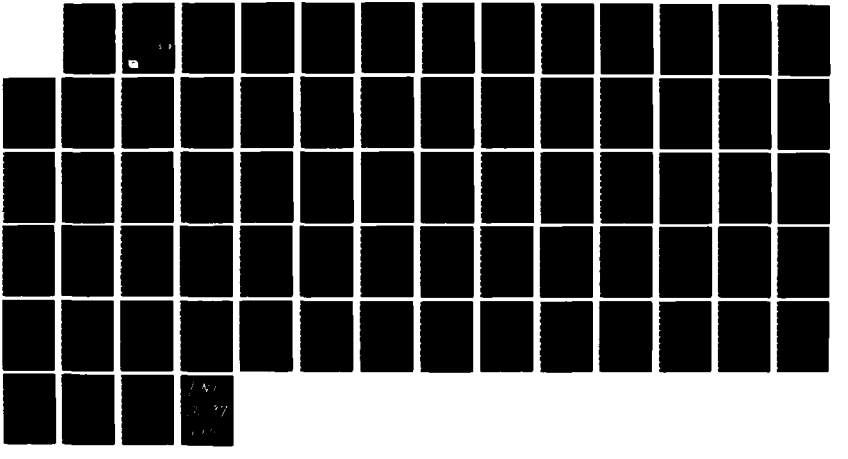
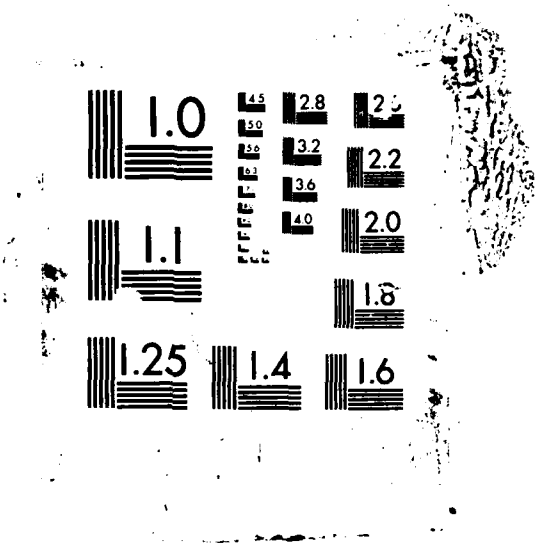
AD-A176 154 THE DEPARTMENT OF DEFENSE REQUIREMENTS FOR ENGINEERING 1/1
INFORMATION SYSTEM (U) INSTITUTE FOR DEFENSE ANALYSES
ALEXANDRIA VA J L LINN ET AL JUL 86 IDA-P-1953-VOL-2
UNCLASSIFIED IDA/HQ-86-31351 MDA903-84-C-0031 F/G 5/2 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.8
3.2
3.6
4.0

(2)

IDA PAPER P-1953

THE DEPARTMENT OF DEFENSE REQUIREMENTS FOR ENGINEERING INFORMATION SYSTEMS (EIS)

Volume II: Requirements

Joseph L. Linn, *Editor*
Robert I. Winner, *Editor*
EIS Requirements Team

July 1986

Prepared for
VHSIC Program Management Office

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311

IDA Log No. HQ 86-31351

The work reported in this document was conducted under contract MDA 903 84 C 0031 for the Department of Defense. The publication of this IDA Paper does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that agency.

This Paper has been reviewed by IDA to assure that it meets high standards of thoroughness, objectivity, and sound analytical methodology and that the conclusions stem from the methodology.

REPORT DOCUMENTATION PAGE

AD-A176154

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Public release/distribution unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) P-1953	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Institute for Defense Analyses 6b OFFICE SYMBOL IDA	7a NAME OF MONITORING ORGANIZATION
6c ADDRESS (City, State, and Zip Code) 1801 N. Beauregard St. Alexandria, VA 22311	7b ADDRESS (City, State, and Zip Code)
8a NAME OF FUNDING/SPONSORING ORGANIZATION VHSIC Program Office 8b OFFICE SYMBOL (if applicable) VPO	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MDA 903 84 C 0031
8c ADDRESS (City, State, and Zip Code) 1211 Fern St., C-112 Arlington, VA 22202	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. T-D5-364 / WORK UNIT ACCESSION NO.

11 TITLE (Include Security Classification)
The Department of Defense Requirements for Engineering Information Systems (EIS). Volume II: Requirements.

12 PERSONAL AUTHOR(S)
Joseph L. Linn, Editor; Robert I. Winner, Editor; EIS Requirements Team

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Final	FROM _____ TO _____	1986 July	72

16 SUPPLEMENTARY NOTATION

17 COSATI CODES		
FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Very High Speed Integrated Circuits (VHSIC), Engineering Information Systems (EIS), microchips, specifications, prototypes, tools, interfaces, networks, operating systems, digital electronic systems, environments, OCREIS, requirements

19 ABSTRACT (Continue on reverse if necessary and identify by block number)
IDA Paper P-1953 comprises two volumes: Volume I, Operational Concepts, and Volume II, Requirements. Volume I contains full set of requirements for the Engineering Information Systems (EIS). Both short-term and long-term requirements are provided as well as a general discussion of the operational concepts and objectives of the EIS program. This document, Volume II, lays out the requirements of the EIS, both for the prototype EIS and for the long term. The first section provides a global introduction to the EIS concept; it is identical to the first section of the first volume. The second section details EIS prototype demonstration requirements. The third section presents the core short-term requirements. The fourth section presents what are called the "extended" short-term requirements; these requirements are not mandatory for the short term but are considered to have very high priority in the longer term. The fifth section presents other long-term requirements.

Volume I , Operational Concepts, provides the definitions and terminology that used used throughout the two volume set.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL	22b TELEPHONE (Include Area Code) / 22c OFFICE SYMBOL

IDA PAPER P-1953

THE DEPARTMENT OF DEFENSE REQUIREMENTS FOR ENGINEERING INFORMATION SYSTEMS (EIS)

Volume II: Requirements

Joseph L. Linn, *Editor*
Robert I. Winner, *Editor*
EIS Requirements Team

July 1986

INSTITUTE FOR DEFENSE ANALYSES

Contract MDA 903 84 C 0031
Task T-D5-364

Table of Contents

1 **Introduction** 1-1
1.1 EIS Program Goals 1-2
1.2 Background 1-5
1.3 Purpose of This Document 1-6
1.4 Organization of This Document 1-7
Last Page of Section 1-7

2 **Prototype Demonstration Requirements** 2-1
Last Page of Section 2-2

3 **Short-term Requirements** 3-1
3.1 Tool/Workstation Integration 3-1
3.2 Data Exchange 3-4
3.3 Engineering Management and Control 3-8
3.4 Information Management 3-14
3.5 External System Interfaces 3-17
3.6 Object Management System 3-18
3.7 Distributed Operating System Facilities 3-19
3.8 Distributed Data Management Facilities 3-20
3.9 Engineering Information Model 3-24
3.10 Rule Processing 3-26
3.11 Control Points 3-27
3.12 User Interface 3-27
3.13 EIS Administration 3-29
3.14 Programmatic Interfaces 3-32
3.15 Design and Implementation Requirements 3-33
Last Page of Section 3-34

4 **Extended Short-term Requirements** 4-1
4.1 Tool/Workstation Integration 4-1
4.2 Data Exchange 4-1
4.3 Engineering Management and Control 4-1
4.4 Information Management 4-1
4.5 External System Interfaces 4-2
4.6 Object Management System 4-2
4.7 Distributed Operating System Facilities 4-2
4.8 Distributed Data Management Facilities 4-2
4.9 Engineering Information Model 4-2
4.10 Rule Processing 4-2
4.11 Control Points 4-3
4.12 User Interface 4-3
4.13 EIS Administration 4-3
4.14 Programmatic Interfaces 4-3
4.15 Design and Implementation Requirements 4-3
Last Page of Section 4-3

5 **Long-term Requirements** 5-1
5.1 Tool/Workstation Integration 5-1
5.2 Data Exchange 5-2
5.3 Engineering Management and Control 5-3
5.4 Information Management 5-7
5.5 External System Interfaces 5-9
5.6 Object Management System 5-10
5.7 Distributed Operating System Facilities 5-10
5.8 Distributed Data Management Facilities 5-11
5.9 Engineering Information Model 5-11
5.10 Rule Processing 5-12
5.11 Control Points 5-12
5.12 User Interface 5-13
5.13 EIS Administration 5-14
5.14 Programmatic Interfaces 5-16
5.15 Design and Implementation Requirements 5-17
Last Page of Section 5-17

1. Introduction

The complexity of engineered systems is increasing dramatically. Advances in the miniaturization of electronics have increased the complexity of electronic designs by an order of magnitude within the last few years. Further, the advent of VHSIC technology promises to increase the complexity of single-chip designs by another order of magnitude.

The complexity of current systems already is so great that it would be practically impossible to carry out the engineering process without computer assistance. Thus, many different tools and support systems for computer-aided design (CAD), computer-aided manufacturing (CAM), computer-integrated manufacturing (CIM), and (generically) computer-aided engineering (CAE) have been evolved and continue to be introduced. Since these tools essentially shoulder a portion of the complexity of an engineered system, the amount of complexity that engineers must bear is substantially reduced.

Yet, the usefulness of these tools and systems is reduced in the current situation since no particular vendor has an integrated tool set that performs all of the steps needed and/or desired for engineering a system from the requirements phase, through specification and design, all the way to manufacturing and maintenance. Thus, the creation of an adequate tool set requires that tools from different vendors be integrated into the tool set. The problem here is that tools from different vendors utilize different models and formats for representing the same information in a design. Different user interfaces are employed; similarly, different approaches are used in implementing administrative and management capabilities. These differences create additional complexity in the engineering process; the elimination of this complexity would allow the potential productivity gains from CAE tools to be more fully realized.

A second problem brought about by the increased complexity of engineered systems is that it is no longer possible in most cases for a single engineer to design the entire system. Rather, complex designs must be subdivided into smaller units and the designs must be handled by design teams rather than by individual designers.

The decomposition of a design often creates highly interrelated subtasks that must be pursued concurrently, yet the designers must use or revise each other's results. Thus, there is a need for controlled sharing of the design information, tracking of design information, tracking of design dependencies and changes, and monitoring of the design process. In short, there is a need for a system that provides database management functions for engineering information.

In response to these problems, an Engineering Information System (EIS) was conceived that provides a framework for tool integration based on information sharing. In this document, the term **EIS** is used to describe the particular system whose concepts and requirements are defined in this document and not in a generic way. Like an operating system, the EIS offers facilities and defines interfaces to be used by applications. Also like an operating system, the EIS

controls and allocates resources (here, data resources), provides concurrency controls, archiving, and an *ad hoc* query capability.

The basic functions of the EIS are:

- Tool Integration--the ability to operate, efficiently and uniformly, a number of tools with different data and hardware requirements.

- Data Exchange--the ability to translate and to communicate data among different hosts and tools not only within the EIS but also between the EIS and external systems (including other EISs).

- Engineering Management and Control--the facilities to monitor the design process and to impose automatic and manual controls on accessing and modifying data.

- Information Management--facilities to describe and to control globally available EIS data (including the creation and manipulation of data, the imposition of data validation (or constraint) checking, the management of versions and of configurations, the control of concurrent transactions, and the management of backup and archived data).

- EIS Administration--the tools and the specifications for managing the data dictionary, tools, workstations, user profiles, and control rules.

The EIS is a set of services and specifications; it is **not** a single implementation of a specific system; rather, it is a framework for providing the necessary functions. It is intended to apply to a number of different engineering tasks and organizations. Therefore, it must offer a means for tailoring to meet the individual requirements of each. It must be able to evolve to meet the challenges of new design processes and tools, and it also must be able to function efficiently in a distributed environment that includes different types of mainframes and workstations. Finally, the system itself must be portable across different systems.

1.1. EIS Program Goals

The EIS program pursues two primary objectives. The first is to produce a set of reference specifications for use by industry that can form the basis for a standardization effort and that covers various data interchange and tool portability aspects. The second is to produce a prototype system that demonstrates how short-term requirements for an EIS can be satisfied using those reference specifications. This section discusses the goals that underlie the two objectives.

The EIS framework consists of a set of fundamental services, much like an all-purpose operating system, and a series of specifications, forming a baseline for

communication and implementation. The goals of the EIS, then, are to provide the services and specifications that are essential to build information systems to support computer-aided engineering design. Furthermore, these services and specifications must be acceptable to a wide body of industrial and government vendors and users.

Integration and Uniformity

The EIS is intended to integrate the tools and processes, engineering and managerial, used in the design environment, and to allow different organizations (teams, departments, corporations, or agencies) to exchange pertinent information. These goals can be stated more specifically as:

(1) Provide a framework for integrating design tools in a cost effective manner.

The EIS will not be a complete, self-contained design automation system; instead it must provide mechanisms (e.g., services and specifications) that facilitate the integration of new and existing tools requiring the least possible modification of the tools themselves. This acknowledges the fact that there is no single, small set of tools that covers all design aspects and activities and is acceptable to all design engineers.

(2) Encourage the portability of tools.

The EIS cannot mandate or ensure the portability of tools among different run-time and design management environments, since it is not itself a tool provider. However, the EIS can provide services and specifications that, when utilized, greatly improve tool portability without greatly restricting tool applicability. Use of these services and specifications should result in cost savings to tool builders as an additional incentive.

(3) Encourage a uniform design environment.

The design environment is implied by the operating environment that the tools create for its users, for example, through the methods of interfacing with the end-user. A frequent reason for lack of uniformity is the absence of a suitable reference specification for user and system interface services that are portable among host environments. The EIS should provide specifications and services that can serve as the basis for significant improvements in this area.

(4) Facilitate the exchange of design information.

Design data is exchanged among different physical processing components, among different software tools, among different engineering departments in the same company, and among different corporations and agencies. The EIS should contain facilities that can be used to provide a uniform tool environment, tools that can perform generic

application services (for example, design format translators), and specifications that can serve as a reference for data exchange.

(5) Provide a framework for supporting design management and reuse of previous designs.

Large designs typically are performed by engineering teams. This requires, for example, controlled sharing of preliminary design data, protection of released design information, and monitoring of design methods and progress. Also, with the rapid accumulation of design data, the reuse of past designs is becoming increasingly dependent upon support from automated tools. An EIS that does not offer design management support would be incomplete.

Acceptability

There is a wide-spread desire for integrated design environments throughout the industry. If this desire led to the development of many incompatible engineering information systems, many of the primary goals expressed above would not be achieved. Hence, a critical factor for the success of the EIS can be expressed as the following goal:

(6) Achieve wide-spread acceptance of the EIS by the electronics industry.

Extra effort may be needed for achieving acceptance on a large scale. For example, the EIS must appeal to end-users as well as decision makers. In addition to demonstrating its usefulnesss and cost-effectiveness, the EIS must also exhibit the key properties of being adaptable and extensible.

Adaptability and Extensibility

In order for the EIS to be widely accepted, it must be able to adapt to the changing needs of the different organizations over time. Its scope must be broad enough, and its generality great enough, to accommodate and integrate new engineering activities. This observation leads to the formulation of the following two goals:

(7) Be adaptable to future changes in engineering methods.

The EIS architecture must be technology-independent, where functional components can be easily replaced, without compromising other system features such as ease of use.

(8) Although concentrating initially on electronic design, be extensible to all life cycle activities and other engineering disciplines.

Because the EIS is a framework, it can be extended to cover more engineering applications by incorporating new or different tools and expanding the scope of data that it manages.

Evolutionary Approach

It will be necessary to insert the EIS into existing design environments. A revolutionary approach is likely to fail for cost reasons. This leads to the formulation of another key constraint as an EIS goal:

(9) Provide a transition path for existing design environments that is cost-effective.

The EIS needs to offer means for a phased transition, offering services that facilitate and reduce the cost of the transition process--including adaptation of design and management tools, acquisition and installation of needed system services and resources, definition of management procedures, transition of existing data, and training of future system users.

1.2. Background

Computer-aided engineering (CAE) systems have evolved in such a way that there is considerable overlap in the functions of the tools provided in such systems. Because of this overlap, it can be very difficult to get the tools to work together. The objective of this program is to provide a framework for the integration of CAE tools and systems. As the development schedule for the project is very ambitious, success may be achieved only by building upon the extensive existing base of design automation tools and systems. Since EIS environments must build from the existing engineering support environments, it is critically important to determine how current nonintegrated systems may evolve to participate in an integrated EIS environment.

1.2.1. Project Organization

The EIS effort was initiated through the efforts of the VHSIC (Very High Speed Integrated Circuits) Program Office. There is a great need for this work in industry; however, no successful unified set of standards, de facto or otherwise, has emerged. Interested parties include the IEEE, the ACM, the ASME, numerous industry concerns, and the United States government.

The technical project manager is Capt. Anthony Gadient, AFWAL/AAL, Wright-Patterson Air Force Base. Questions about the EIS program should be directed to him.

1.2.2. Project History

A large amount of work has gone into automating various areas of engineering, especially electrical engineering. This program was initiated to search for a

means to integrate existing engineering systems and tools. At first, it was thought that this is essentially a database problem; as it turns out, the scope of the problem is far greater. In defining an EIS environment, the issues to be considered include inter-tool interfaces, network and operating system interfaces, consistent user interfaces, standardization efforts and policies, and the migration of existing tools and systems into an integrated EIS environment. In addition, the concept of extensibility is a pervasive influence on all other system concepts.

A fundamental tenet of the approach adopted in this program is that the environment issues that the EIS is intended to address can be considered separately from the tool set that is to populate the environment. Thus, the solutions obtained should be applicable across wide ranges of tool sets; that is, tool sets from various engineering disciplines should be equally appropriate in an EIS environment. However, the prototype EIS will explore only the digital electronics field. Similarly, there are many facets of the design of digital electronic systems; the prototype EIS will concentrate on the design of such systems.

1.3. Purpose of This Document

This document contains a full set of requirements for the EIS. Both short-term and long-term requirements are provided as well as a general discussion of the operational concepts and objectives of the EIS program.

This document is the latest step in a program aimed at formulating requirements for the EIS. Previously, workshops were held in November, 1985, and January, 1986. In the first workshop, the participants were divided into two working groups in order to identify technical issues concerning the development of standards and the prototype EIS. The workshop identified eight overlapping areas of concern including: Current Representation Capabilities, Man-Machine Interfaces, Computing Systems, Information Management and Control, Information Models, Functional Architecture, Workstations, and Application Program Information Requirements.

In preparation for the January workshop, the Institute for Defense Analyses (IDA) prepared a document entitled "Operational Concepts and Requirements for an Engineering Information System" (OCREIS)." The OCREIS document served as a starting point for discussions in the working groups of the January workshop. The working groups were chartered to provide input to the next documents. These comments were important in reformulating the original OCREIS document to obtain the next document, OCREIS version 2.

OCREIS version 2 was submitted to a qualified panel of experts and to the attendees of the previous workshops for consideration and comment. Based on these new comments and a parallel requirements analysis effort, IDA produced the next document, OCREIS version 3. Another expert panel (actually a superset of the previous panel) reviewed the third version of the document.

Their comments and an ongoing requirements conflict analysis has yielded the current, final version of the document, the one that wil' be operative for the EIS prototype development.

1.4. Organization of This Document

This document is organized into two volumes. The first volume, **Operational Concepts**, provides the definitions and terminology that are used throughout the two-volume set. Since a number of terms that are generic in general usage take on very specific meaning in the EIS, the reader is cautioned to bear in mind the terminology found in the first volume. The first volume is organized as follows. The first section provides a global introduction to the EIS concept. The second section provides a high level summary of the EIS functionality. The third section considers a number of problems to be addressed in the EIS environment, defines a number of relevant terms, presents a conceptual architecture for the system, and provides a rationale for the organization of the requirements. There are two appendices; Appendix A contains a glossary of terms, and Appendix B contains a listing of relevant literature

The second volume, **Requirements**, lays out the requirements of the EIS, both for the prototype EIS and for the long term. The reader is cautioned again to bear in mind the terminology found in the first volume. The second volume is organized as follows. The first section provides a global introduction to the EIS concept; it is identical to the first section of the first volume. The second section details the EIS prototype demonstration requirements. The third section presents the core short-term requirements. The fourth section presents what are called the "extended" short-term requirements; these requirements are not mandatory for the short term but are considered to have very high priority in the longer term. The fifth section presents other long-term requirements.

2. Prototype Demonstration Requirements

Following requirements formulation, the next step of the plan adopted for the development of EIS technology is the development of a prototype of the EIS software and specifications. In keeping with the goals of the EIS program and of the VHSIC program, the prototype is to be developed with the following goals:

- to provide a convincing demonstration of the feasibility and efficacy of the EIS approach.
- to provide a basis for standardization of important information and control interfaces relating to electronic design environments.
- to provide a baseline for the continued evolution of EIS technology.
- to provide a vehicle for integration of VHSIC technology tools.

There are many different ways that the prototype system could be developed and demonstrated in accordance with these goals. Thus, the prototype demonstration goals are minimal.

2.1. The prototype demonstration must be capable of **demonstrating** the implementation of each of the short-term requirements.

2.2. The prototype demonstration environment must satisfy each of the following properties:

(a) it must contain workstations from at least two different vendors.
(b) it must contain at least two native architectures (examples of native architectures are a DEC VAX, an IBM System/370, a Motorola 68000) and the order of byte-storage relative to word-storage of one architecture must be different from another architecture in the environment.
(c) it must contain a data communications network.
(d) it must contain systems running under at least two operating systems.
(e) it must contain at least two integrated systems.
(f) it must contain at least one attached system.

2.3. The tool set used in the prototype demonstration must contain a reasonable complement of tools that use VHDL as their primary input/output format.

2.4. The tool set used in the prototype demonstration must contain both integrated and attached tools.

2.5. The prototype demonstration environment must be instrumented to determine the performance effects of running tools under the EIS as opposed to running them without the EIS.

2.6. As part of the demonstration, information must be available detailing how the prototype development conforms with requirement 3.14.1.2 (compatibility with the Common APSE Interface Set), requirement 3.14.6.1 (all programmatic interfaces callable from Ada*), and each of the requirements in Section 3.15 (Design and Implementation Requirement).

*Ada is a registered trademark of the U.S. Government, Ada Joint Program Office.

3. Short-Term Requirements

3.1. Tool/Workstation Integration

3.1.1. Tool Initiation

3.1.1.1. The EIS must provide a capability that permits client-initiated and automatic invocation of any integrated or designated attached tool (interactive or noninteractive) that is available on the local host. In addition, the capability must permit invocation of any integrated or designated attached noninteractive tool remotely, including tools that operate in a host environment that is different from the host environment of the client.

3.1.1.2. As part of this capability, the EIS must:

(a) determine if copies of the objects required as parameters for tool initiation are accessible at the host designated for the tool execution.
(b) determine whether the necessary design libraries are available.
(c) if necessary and permissible, transfer the required objects to a data repository accessible at the tool's host (including the tool itself).
(d) cause the tool to be executed.
(e) return the result objects, errors, status, etc., from the tool execution to the client-specified destinations.

3.1.1.3. The EIS must be able to select, automatically and under client direction, an appropriate host for executing the tool, based on resource requirements, availability, cost, and performance factors.

3.1.1.4. The EIS must support deferred and immediate invocation of tools, including the serial and parallel execution of tools. The EIS must allow the client to specify whether separate tool invocations must be executed serially or can be executed in parallel.

3.1.1.5. The EIS must support piping of data (i.e., using an output of one tool invocation as input to another serial or parallel tool execution, including the ability to concatenate several tool outputs into a single tool input).

3.1.2. Parameter Handling

3.1.2.1. When a tool is invoked by a user, the EIS must be capable of executing a user dialog that may be required for initiation of the tool. The dialog requirements include, for example:

(a) querying the user to obtain any required parameters.
(b) guiding the user through the dialog for defining the parameters.
(c) using EIS administrative data to specify default values for tool parameters.

(d) specifying any piping of data that may be required.
(e) allowing the user to bypass all or part of the dialog by providing command input.

3.1.2.2. The EIS must use the administrative data it manages (see Section 3.13) for determining the parameters needed for invoking a tool.

3.1.2.3. The EIS must verify, as much as possible, that the characteristics of the supplied parameters (after any format translations) are consistent with the parameters specified in the tool description (discussed in more detail in Section 3.13).

3.1.3. Execution Control

3.1.3.1. The EIS must provide an alert mechanism for notifying clients of the completion of the background execution of a tool and of intermediate milestones defined by the tool.

3.1.3.2. The EIS must provide a mechanism for communicating to the client any error messages that are generated by the execution of a tool, and for providing notification to the client in case of abnormal termination of a tool.

3.1.3.3. The EIS must recognize the normal or abnormal termination of a tool client or of a user interface, offer a mechanism for informing tools that were initiated by the terminated client about this event, and permit background executions initiated on behalf of the user or tool client to proceed without impact and foreground executions to be terminated.

3.1.3.4. The system must allow the current status of a tool execution to be queried by an appropriately authorized client.

3.1.3.5. The system must allow termination of any tool by an appropriately authorized client.

3.1.3.6. Generally, clients that initiated a tool would be considered authorized to query its status or to terminate it, but this may not hold true for all situations.

3.1.4. Data Setup and Disposition

3.1.4.1. The EIS must be able to invoke automatically all data exchange services that are required in the context of a tool invocation, to the extent that the EIS supports such services (see Section 3.2).

3.1.4.2. Repeated tool execution should not require repeated format conversion. It is not a requirement that the results of a format conversion be saved by the system; it is a requirement that if that format conversion is costly to execute, then a mechanism is supplied which prevents that format conversion from having to be repeatedly executed.

3.1.5. Automated Program Networks

3.1.5.1. The EIS must support the execution of completely automated networks of programs that specify a sequence of serial and parallel execution of tools and other programs for noninteractive execution.

3.1.5.2. The EIS must provide facilities that permit users to add, delete, modify and display automated program networks.

3.1.5.3. Automated program networks must support the tool integration requirements specified earlier, such as the specification of serial and parallel execution of programs, the routing of input/output data (including piping of data among programs), and allowing invocation by users and programs.

3.1.5.4. The EIS must support the use of parameters in the definition of program networks, and support the instantiation of the parameters when a program network is invoked, analogously with the invocation of tools.

3.1.5.5. Automated program networks must allow for the specification of decision points, and must make the decisions dependent upon the values of program network parameters, success or failure of program execution, and values of state variables that are accessible to programs executed by the program network.

3.1.5.6. Automated program networks must allow for the specification of messages to be sent to the invoking user.

3.1.5.7. The EIS must provide the user with capabilities for monitoring the execution of automated program networks that are analogous with the tool monitoring capabilities, such as inquiring the status, receiving error messages, and being notified of completion or abnormal termination of the program network.

3.1.6. Workstation/Host Interface

3.1.6.1. The EIS must provide the capability for a host to accept requests from clients outside the host, for the execution of tools residing at that host.

3.1.6.2. The EIS must provide the capability for a host to accept requests from clients outside the host, against system services available at that host. Examples include use of attached devices or the local data store.

3.1.6.3. The EIS must permit an authorized user to prevent requests from outside a host to be serviced by the host.

3.1.6.4. The EIS must permit a host to support the preceding capabilities without requiring that EIS components, per se, be resident on the host. For example, an organization must be able to provide a custom-written program that

responds to EIS-specified protocols and provides the required capabilities through host-specific means.

3.1.6.5. A host must be capable of continuing operation even when the interface to the rest of the EIS is not functional. Continued operation must be possible as long as the design/engineering data and the design/engineering tools that are needed to continue operation are local to the host and as long as no authorization from a source external to the host is required. This means, for example, that a workstation must not be forced to make use of external administrative or control functions in order to provide access to local tools and local data.

3.1.6.6. Where possible, access to services on remote hosts must be supported as long as these hosts and services are operational, regardless of the availability of other hosts. For example, directing graphic output from a workstation to a host controlling the needed graphic output device should not require the use of administrative and control functions on a third host.

3.1.7. Version/Configuration Management for Tools

3.1.7.1. The system must provide version and configuration management capabilities for tools. These requirements are presented in a more general context in Section 3.3.

3.1.7.2. The addition of new tools and the replacement of existing tools must not have any uncontrolled impact on currently executing operations, and must not interrupt the system or otherwise make tools unavailable for an extended period of time. For example, the EIS must provide the ability to determine which currently executing operations or automated program networks reference a tool that is being replaced, and to terminate these operations or let them complete with the old tool version.

3.1.8. Tool Back-up

No short-term arequirements.

3.2. Data Exchange

3.2.1. General Data Exchange Requirements

3.2.1.1. The EIS must provide a service, available to all clients, that supports the transfer of design data objects for the purpose of moving and copying them among system services (e.g., integrated and attached data repositories), program input/output streams, and EIS system components.

3.2.1.2. A client must be able to direct the EIS to perform the data exchange operation in the following modes:

(a) in foreground
(b) in background, with immediate initiation of the transfer
(c) in background, with deferred initiation of the transfer

3.2.1.3. The EIS must provide an alert mechanism for notifying client(s) of the completion of a deferred transfer operation.

3.2.1.4. A client must be able to transfer any number of data objects with a single data exchange operation, and to specify the data objects to be included by reference or by database query, interactively or programmatically. The EIS must be able to disassemble the exchanged data into its constituents when necessary.

3.2.1.5. The EIS must be able to transfer any type of data object. No assumptions may be made about restrictions on the type of engineering or management information that will be exchanged.

3.2.1.6. The EIS must not assume that all data exchange is routed through a central facility or uses an integrated data repository as the means for exchanging data. However, a user organization must be able to direct the EIS to transfer information only via an integrated data repository or some central facility.

3.2.2. Data Exchange Formats

3.2.2.1. The EIS must define and support a format for data exchange (called the **common exchange format**).

3.2.2.2. It must be possible to represent in the common exchange format an arbitrary object such that it can be reconstructed in a fashion that retains its structure, and also preserves the semantics of any type of component that is defined in the Engineering Information Model (EIM), to the extent that the semantics are captured in the EIM (see Section 3.9).

3.2.2.3. The common exchange format may use the modeling language of the Engineering Information Model directly, or may be a combination of several generally-accepted design representation languages. Where the EIM modeling language is not used directly, there must be a one-to-one mapping defined between the EIM and the common exchange format that specifies the relationship between the information types in the EIM and how information is represented in the common exchange format. It is desirable to minimize the number of design representation languages used in the construction of the common exchange format.

3.2.2.4. The common exchange format must be extensible by an organization in a prescribed fashion that permits identification of such extensions and that

permits programs to determine syntactically when extended information is present and to ignore it.

3.2.2.5. The method of extension must ensure that an interpretation that ignores the extensions yields a description that is valid and complete with respect to the common exchange format definitions.

3.2.2.6. The method of extension must ensure that user extensions will not conflict with future extensions of the common exchange format.

3.2.2.7. All EIS-provided programs that interpret the common exchange format must provide a mechanism for executing custom programs that can deal with user extensions.

3.2.2.8. The common exchange format must employ an encoding that is based on a printable character set that is common to ASCII and EBCDIC, and that ensures a machine-neutral representation of the information except for character-representation issues.

3.2.3. Data Exchange Support Requirements

3.2.3.1. If more than one design representation language is used in the construction of the common exchange format, there may be several equivalent schemes for encoding and structuring information. If so, the EIS must provide means for automatically translating between the different schemes.

3.2.3.2. If equivalent semantics can be expressed in more than one design representation language used for the common exchange format, the EIS must provide a means for translating among them.

3.2.3.3. The EIS must provide programs for editing information stored in the common exchange format. The editors must be able to deal with user extensions to the common exchange format.

3.2.3.4. The EIS must provide programs for translating between VHDL/EDIF and the common exchange format for information that can be represented in the common exchange format, to the extent that VHDL/EDIF is not already the chosen common representation for this information.

3.2.3.5. The EIS must be able to invoke interactive and noninteractive programs for translating among different methods of design representation and hardware-dependent encoding of data, and for extracting a subset of the information represented by a data object. This includes the ability to invoke programs for translating among common exchange formats, and between common exchange formats and other formats.

3.2.3.6. The EIS must provide for the exchange of information via electronic data communication. If an automated transfer mechanism between two hosts exists, the EIS must support an end-to-end data exchange service in which the

mechanism for the transfer is transparent to the end-user. The EIS must be able to utilize any electronic data communication facility that provides at least ISO level 4 capabilities, including nonautomated, nonelectronic transfers (e.g., tapes).

3.2.3.7. Normally, the EIS must take responsibility for the execution of the data exchange without further client interaction. It is desirable that the EIS provide the data exchange as an atomic operation.

3.2.3.8. The EIS must identify a data exchange operation in a unique fashion (e.g., to permit a client to query its status or for locating it in an audit trail).

3.2.3.9. To the extent possible, the EIS must support the notion of "urgency" of a data exchange (e.g., specified priority or a time limit for completion) that can be specified by the user organization or the requesting client.

3.2.3.10. The EIS must be able to handle the error and exception situations that may occur while a data exchange operation is executed or waiting to be executed, and that affect the successful completion of the operation (i.e., that cannot be dealt with in a transparent fashion by lower-level components). For on-line operations, the EIS must provide well-defined error indications to the invoking client. For background operations, the EIS must provide an alert mechanism. For example, the EIS must monitor the operations against the specified transfer requirements and alert specified clients if they cannot be met (e.g., if a deferred data exchange is not completed within the specified time frame).

3.2.4. Exchange of Management Data

3.2.4.1. For data that are transferred using the common exchange format, the EIS must be able to include automatically administrative information related to the transferred design data object. A user organization must be able to direct the EIS as to which types of information should be included, depending upon such considerations as the type of source and target data object, origin and destination of the transfer, and current state of the transferred object or any configuration of which it is a member.

3.2.4.2. A client must be able to request additional administrative information to be included with each individual request.

3.2.4.3. The EIS must be able to separate this administrative data from the design data objects when necessary.

3.2.4.4. The EIS must provide programs for converting between administrative information represented in the common exchange format and administrative information stored according to the EIS common database schema (see Section 3.8).

3.2.4.5. The following is a list of the classes of administrative information that should be considered in this context:

(a) data object class and identification
(b) sending organization
(c) sending client
(d) if the sending client is a program, also the sending user
(e) unique identification of the data exchange request, including a time stamp
(f) reference information to support future inquiries or alerts
(g) information about the configuration of the sending EIS
(h) information about the configuration of the object being transmitted
(i) design history information
(j) information regarding audit trail and change control tracking to be performed at the destination for the object being transmitted
(k) comments

3.3. Engineering Management and Control

3.3.1. Object Registration

3.3.1.1. The EIS must be able to maintain, for engineering management and control purposes, information about each user, tool, service, data object, operation, etc. Each such object registered with the EIS is referred to as an ***EIS-controlled object.***

3.3.1.2. Authorized clients must be able to query, in interactive or non-interactive mode, all management and control information maintained by the system. As examples, this information might include, as appropriate to the class of object:

(a) classification
(b) object identification, aliases and version
(c) revision level
(d) time of creation
(e) identification of the client who created the object
(f) role (e.g., role of a **user**, where user is a class of object)
(g) accounting information, such as project id and account number
(h) design release status
(i) security classifications, such as authorizations and protection levels
(j) required run-time environment capabilities
(k) required user interface capabilities
(l) permissible input object classes
(m) classes of output objects
(n) operations permitted against the object
(o) keyword descriptors

3.3.2. Configuration Management

3.3.2.1. The EIS must be able to associate with an object all of its component objects, plus related objects as directed by the user organization. This description is called the **configuration** of the object. Configuration definitions will vary depending on the object class (for example, design object, tool, EIS itself) and must be able to include any groupings of:

(a) component decompositions
(b) component representations
(c) alternatives for any component, subcomponent, or representation
(d) object versions
(e) versions of any component, subcomponent, or representation
(f) parameters used to instantiate the object
(g) tools and parameters used in the creation of any component, subcomponent, or representation
(h) related design library objects
(i) related derived or source objects
(j) related programs and documents

3.3.2.2. An authorized client must be able to define, delete, and modify configurations. For example, the ability to define a configuration includes the ability to construct the configuration by executing a database query.

3.3.2.3. It must be possible for different configurations of an object to include the same subordinate object. The system must prevent loops in a configuration description.

3.3.2.4. A configuration must be able to refer to other configurations among its subordinate objects.

3.3.2.5. The EIS must be able to identify all members in a given configuration, and all configurations of which an object is a member.

3.3.2.6. The EIS must support partial configurations (i.e., configurations in which not all subordinate objects are fully specified). For example, a user must be able to define and refer to partial configurations.

3.3.2.7. The system and clients must be able to determine that a configuration is partial.

3.3.2.8. When a subordinate object of a configuration is modified, the EIS must be able to determine whether to create a new configuration, modify the old configuration, or take no action on the configuration definitions. The EIS must be able to make this determination based on explicit (**ad hoc**) direction from the client or on default specifications.

3.3.2.9. The EIS must be able to recognize that a member of a configuration has been changed and that its new version is not included in the configuration.

3.3.2.10. In addition to the requirements outlined here, management and control of configurations must be subject to all other requirements identified in this section for EIS-controlled objects in general. In particular, the management and control of configurations must also satisfy the requirements for:

(a) dependency tracking
(b) change control
(c) version control
(d) security and access control

3.3.3. Dependency Tracking

3.3.3.1. The EIS must be able to establish dependencies among objects based on *ad hoc* or default specifications, inputs and outputs of tool execution, or data exchange operations.

3.3.3.2. The EIS must be able to maintain information about dependencies. This information must be able to include:

(a) identifier (including version number) of the dependent object.
(b) identifiers (including version number) of the object(s) on which the object depends.
(c) identifier (including version number) of the client that established or validated the dependency.
(d) identification of the operation that created or validated the dependent object.
(e) time that the operation began/completed.
(f) classification of the dependency.

3.3.3.3. Both the system and authorized clients must be able to read and change the dependency information for an object. The system will use this information for change control, in particular.

3.3.3.4. The EIS must be able to compute the transitive closure of the dependency information (i.e., to determine the entire chain of indirect dependencies of one object on another). In this computation, the system must be capable of distinguishing, as directed by a client, among the various classes of dependencies.

3.3.4. Change Control

3.3.4.1. The EIS must be able to recognize attempts to change controlled objects. In particular, the EIS must be able to recognize:

(a) attempted and completed changes to objects, including changes to associated management and control information.

(b) performance of controlled operations.
(c) time-dependent events, such as a specific time having passed without an expected operation being performed.

3.3.4.2. The EIS must be capable of making decisions regarding change control actions that should be taken. The EIS must be able to make the decisions dependent upon the nature of the change, and upon management and control information associated with the object to be changed, dependent objects, and configurations of which the object is a member. The support for change control actions must include:

(a) denying the change.
(b) alerts (e.g., sending messages to clients).
(c) changing management and control information associated with the object, dependent objects, and configurations of which the object is a member.
(d) invoking a user-specified program that provides facilities other than those above.

3.3.4.3. A client must be able to specify the effective date of a change. The EIS must support a distinction, according to *ad hoc* or default specifications, between change control actions and rules that apply prior to this date and those that apply at the time of effectivity. For example, a user may delay effectivity of a change in order to limit access to the changed data until the changes can be synchronized across installations. In this context, the EIS must be able to recognize that the engineering operation that posted the change has not ended until the date of effectivity.

3.3.4.4. The EIS must be able to determine that other objects upon which a controlled object depends have been changed; this must result in validating, and modifying as necessary, the dependency information for the controlled object.

3.3.4.5. In executing change control actions, the EIS must perform change control recursively on the objects being affected by the actions.

3.3.4.6. The EIS must be able to apply a designated ordering upon the objects affected by change control actions, and to prevent loops.

3.3.5. Version Control

(See also Section 3.4, Information Management, for requirements pertaining to automatic versioning of EIS-managed data objects.)

3.3.5.1. Where possible, there must be a mechanism that allows the EIS to verify that the correct version of each object (e.g., a tool, an electronic document) in the system is used whenever that object is accessed through the EIS.

3.3.5.2. A client must be able to register new versions of objects and define alternative objects. The EIS must check the monotonic ordering of version identifications if provided by the client. Each version and alternative of an object must be treated as an object in its own right.

3.3.5.3. A client must be able to obtain a version history, including a list of the versions corresponding to revision levels.

3.3.5.4. A client must be able to obtain a list of all alternatives of an object.

3.3.5.5. A client must be able to explicitly reference a specific version or alternative of an object when performing an operation (e.g., in constructing a configuration or when invoking a tool).

3.3.5.6. If a client does not explicitly select a version or alternative, the EIS must be able to select a default.

3.3.5.7. The EIS must be able to determine, from management and control information and the class of operation to be performed, whether the operation is valid on the selected version or alternative.

3.3.5.8. A client must be able to read the version stamp of an object, and authorized clients must be able to change it.

3.3.6. Security and Access Control

3.3.6.1. The EIS must provide means for authenticating users, and it must use the management and control information associated with the user for security and access control purposes.

3.3.6.2. The EIS must be able to subject each operation requested through the EIS to access control prior to its execution.

3.3.6.3. The EIS must be able to use, in the execution of the access control, the management and control information associated with any object involved in the operation, any configurations of which these objects are members, and/or any of its dependent objects.

3.3.6.4. The EIS must be able to make a decision about whether to grant or deny the attempted operation, based on the management and control information of the objects involved, and to enforce this decision.

3.3.6.5. The EIS must be able to take specific actions, including:

(a) informing specified clients of an access decision.
(b) if the request is denied, giving the client some indication of the reason for denial.

(c) permitting an authorized user to override an EIS decision, although the system must warn the user that the security and integrity of the data might be compromised.

3.3.6.6. When interfacing with external software services, the EIS must be able to provide the information required to support their access security facilities, and must not support users in bypassing these facilities.

3.3.7. Audit Trail

3.3.7.1. The EIS must be able to capture and maintain a record of all **requested** and **completed** operations.

3.3.7.2. For operations supported by the EIS, capturing must be automatic and transparent to the clients. For other operations, clients must be able to make entries in the audit trail.

3.3.7.3. The information to be captured in an audit trail about a **requested** operation includes, for example:

(a) Information about the operation, for example:
    i. identifier of the client (and user, if different) requesting the operation
    ii. identifiers of all objects referred to in the request (including version stamp)
    iii. operation requested
    iv. date and time of the request
    v. accounting and other administrative information (e.g., project number, account number
    vi. identification of the site/workstation/terminal from which the operation was requested
    vii. in case of manual entry, the identifier and authority of the client making the entry
(b) Information about the system's response, for example:
    i. the system response (i.e., permission granted or denied)
    ii. date and time the response is made
    iii. if permission is granted, the authority by which it is granted
    iv. if permission is denied, the reason for which it is denied

3.3.7.4. The information to be captured in an audit trail about a **completed** operation includes, for example:

(a) Information about the operation (as listed in the previous requirement)
(b) Information about completion of the operation, for example:
    i. date and time of completion
    ii. an indication of whether or not the operation completed successfully

iii. if the operation completed successfully, an explanation of the result and authorization
iv. if the operation failed, an explanation of the type of problem

3.3.7.5. A user organization must be able to specify which types of operations must be audited, and what information must be captured for each type. This includes providing filtering functions for client-provided audit trail entries.

3.3.7.6. Authorized clients must be able to query and to annotate the audit trail. The EIS must not permit any other type of change to the audit trail.

3.3.7.7. The EIS must be capable of combining audit trail information from a distributed environment, in a timeframe consistent with the requirements for using the audit trail information in system decisions.

3.3.8. Design History

No short-term requirements.

3.3.9. Methodology Support

Np short-term requirements.

3.3.10. Authorization and Approval

No short-term requirements.

3.4. Information Management

This section addresses the requirements for supporting the concept of EIS-managed data that include design data objects (e.g., completed designs, designs in progress, and design library modules) and also management and administrative data.

3.4.1. Management of Design Data Objects

3.4.1.1. The EIS must be able to manage design data objects; that is, the EIS must be able to manage an engineering design object as an uninterpreted string. This must not be restricted to design data objects stored in the common exchange format.

3.4.1.2. The EIS must provide a mechanism to protect against loss of data through back-up and recovery mechanisms that are transparent to the clients.

3.4.1.3. The EIS must be able to impose a logically hierarchical organization of at least three levels on the design data objects (e.g., a library system for classifying objects as globally-accessible, project-specific, or private).

3.4.1.4. A user must be able to obtain a complete directory of all objects to which he has access.

3.4.1.5. The EIS must provide data access operations, supported as atomic operations, at least for:

(a) creating an object
(b) retrieving an object
(c) replacing an object
(d) deleting an object

3.4.1.6. In addition, the EIS must be able to support user-definable data access operations on design data objects, for example, schematic extraction and back-annotation.

3.4.2. Administrative and Descriptive Information

3.4.2.1. The EIS must provide the capability to maintain, along with the design data object, descriptive information that can serve as the basis for querying object characteristics. This information includes, for example:

(a) keyword descriptors according to some organization-developed classification scheme
(b) electrical data and test and simulation results, expressed as simple numeric or textual properties
(c) level of confidence
(d) how the object was validated
(e) what tests the object passed
(f) vendor and availability information

3.4.2.2. The EIS must support the specification and invocation of operations for extracting selected information from design data objects, and storing it as descriptive information. In addition, a user must be able to manually enter and update descriptive information.

3.4.2.3. The EIS also must provide the capability to maintain administrative information, that is, all data required for operating the EIS and executing management and control policies (see Section 3.13 for a discussion of EIS administration).

3.4.2.4. The EIS must provide database management support, transparent to the end user, for descriptive and administrative information, including:

(a) atomic transactions
(b) concurrency control
(c) backup, recovery, journaling

(d) data manipulation, including creating, deleting, updating, searching and retrieving
(e) data validity checking (not including consistency checks), according to specifications by the user organization

3.4.2.5. The EIS must automatically synchronize the update of administrative information with operations against the associated design data objects. In addition, the EIS must allow automatic synchronization of mechanisms that maintain descriptive information with operations against the associated design data object.

3.4.2.6. The EIS must support browsing of descriptive and administrative information, where browsing includes the capability to traverse EIS-maintained relationships between objects, as well as relationships based on similar information content.

3.4.2.7. A client must be able to combine descriptive and administrative information in a single query. Any distinction between these two types of information in the underlying data management support must be transparent to clients.

3.4.2.8. The EIS must be able to select and automatically retrieve design data objects as the result of such queries.

3.4.3. Automatic Versioning of Objects

3.4.3.1. When an EIS-managed design data object is replaced, the EIS must be able to automatically create a new version of the object, according to *ad hoc* or default specifications.

3.4.3.2. The EIS must uniquely identify multiple versions of a design data object according to a strictly increasing totally-ordered function. The identification is referred to as the **version stamp**; a unique timestamp or any other identifier that meets the criterion may be used.

3.4.3.3. Versions of an object that are created later must be identified with a higher version stamp than those created earlier. The **latest** version is the one with the highest version stamp.

3.4.3.4. The EIS must be able to generate a new version stamp, higher than that of the latest version, whenever a new version is created.

3.4.4. Support for Concurrent Access to Design Data Objects

3.4.4.1. The EIS must support the definition of engineering operations on design data objects. Data access operations, defined above, are performed in the context of an engineering operation.

3.4.4.2. A user organization must be able to define an engineering operation as conflicting with another engineering operation (or with the same operation).

3.4.4.3. The EIS must be capable of preventing conflicting engineering operations from accessing the same EIS-managed design data object concurrently.

3.4.4.4. The EIS must be able to notify appropriate clients of conflicts.

3.4.4.5. When an engineering operation is allowed to proceed, the EIS must be able to trigger associated actions, such as making backups, sending messages, etc.

3.4.4.6. The EIS must support the definition of shared objects for the purposes of performing engineering operations. The EIS must be able to allow more than one authorized client at a time to operate on a shared data object.

3.4.4.7. When more than one client is operating on a shared object, the EIS must be able to handle any resulting updates according to default specifications (e.g., storing each update to the shared version as different alternatives of the next version number, and/or triggering notifications to other clients about the occurrence of a change in the shared version).

3.4.4.8. The concurrent access controls must not prevent a client from re-issuing a request in the event a design data object is accidentally lost or corrupted.

3.4.5. Interim Modifications to Design Data Objects

No short-term requirements.

3.4.6. Backup and Recovery of Objects

No short-term requirements.

3.4.7. Archiving Objects

No short-term requirements.

3.5. External System Interfaces

3.5.1. Invocation of Tools across EIS boundaries

NO SHORT-TERM REQUIREMENTS

3.5.2. Data Exchange among Different EISs

3.5.2.1. The EIS must provide a mechanism for exchanging design data among different EISs.

3.5.3. Data Exchange with non-EIS Systems

NO SHORT-TERM REQUIREMENTS

3.5.4. Management and Control of External System Interfaces

3.5.4.1. The EIS must be able to subject all data exchange operations that occur via the external system interface to its management and control rules.

3.5.4.2. A receiving EIS must be able to identify a data object as having been received from an external system, and must subject all further local operations on that object to its management and control rules, to the extent that the data is now managed by the receiving EIS.

3.5.4.3. For each object sent to an external system, the sending EIS must be able to identify to which external system the object was sent and which object version was used in the exchange.

3.6. Object Management System

3.6.1. The EIS must provide the object management facilities necessary to support an object-oriented approach. Specifically, the EIS must support the definition of object classes and operations upon objects, provide specifications for the protocols associated with each class, and provide the mechanisms needed to invoke the operations defined for each class.

3.6.2. The definition of the classes and operations must avoid redundancy (i.e., more than one specification for the same concept) and provide consistency (i.e., analogous specifications for analogous concepts). For example, this might be supported through a suitable classification scheme.

3.6.3. The EIS must provide the operations that are needed to support the capabilities required in this document, or be able to interface to external software that implements needed operations. It is desired that the EIS utilize the latter approach to the largest possible extent.

3.6.4. Wherever the EIS uses external software for implementing the required capabilities, the corresponding interfaces must conform to the programmatic interface requirements specified in Section 3.14.

3.7. Distributed Operating System Facilities

The EIS must provide uniform operating system services in a distributed, heterogeneous operating environment.

3.7.1. Queuing and Scheduling

3.7.1.1. The EIS must manage shared resources that are accessible through the EIS, such as devices, accelerators, communication threads, data access service threads, etc.

3.7.1.2. The EIS must be able to queue requests for operations against shared resources where the operation conflicts with the current status of the resource, dequeue requests as resource status changes, and allow authorized users to query and manage the queues.

3.7.1.3. The EIS must provide scheduling services that select among the resources that can be used to satisfy a request based on cost and performance data, and that can select among the queued requests according to priorities and deadlines. User organizations must be able to influence scheduling decisions (e.g., via parameters or user exits).

3.7.1.4. The EIS must be able to cause the execution of programs and service requests across hosts, host environments, and programming languages.

3.7.2. Interconnection Services

3.7.2.1. The EIS must support the establishment of sessions between clients executing on an EIS host and a program executing on the same or a different EIS host. This includes:

(a) a protocol for initiating, terminating, and accepting a session,
(b) uniquely identifying a communication thread for the session that can be used for routing, queuing, and dequeuing messages, and
(c) recognizing abnormal session termination.

The remaining issues (e.g., contents of messages, message protocols) should be the responsibility of the clients that are in session with each other, with the exception of sessions that support EIS services.

3.7.2.2. The EIS must support the ability for a client to engage in more than one session and to perform other activities, concurrently.

3.7.2.3. In addition to establishing sessions, the EIS also must support the exchange of messages in the context of these sessions, including the ability to send a message with or without waiting for a reply and the ability to receive a message.

3.7.2.4. The EIS must provide an alert capability for notifying clients of the occurrence of a specified event (e.g., completion of a background operation, or occurrence of an error or exception situation). Alerts include the capability for specifying the clients to be notified, and for providing immediate notification if a client is currently on-line and deferred notification if a client is currently off-line.

3.7.2.5. A complete set of protocols to support the various communication requirements must be specified. The specifications must use existing protocols that are widely used, where reasonable.

3.7.3. Request Mapping

3.7.3.1. The EIS must provide facilities for automatically mapping requests submitted via a programmatic or user interface into the services needed to execute the request, according to plans specified by a user organization or according to EIS provided mechanisms.

3.7.3.2. The EIS must also provide facilities for collecting the input data needed for executing each of the operations in a plan and routing the output data resulting from their execution.

3.7.4. Process Monitoring

3.7.4.1. The EIS must supervise the coordinated execution of plans. For example, the EIS must be able to synchronize concurrent tasks, and must be able to recognize completion and abnormal termination of tasks.

3.7.5. Error and Exception Handling

3.7.5.1. The EIS must handle error and exception situations that occur while executing any of its services.

3.7.5.2. The EIS must provide well-defined error indications to the client if it cannot correct an error or exception situation.

3.7.5.3. The EIS must attempt recovery from the errors in a fashion transparent to the client where possible.

3.7.5.4. The EIS must be able to support exception and error handling according to *ad hoc* or default specifications.

3.8. Distributed Data Management Facilities

3.8.1. Data Distribution

3.8.1.1. The EIS must permit an organization to distribute physically the data managed by the EIS, while still satisfying the information management requirements stated in Section 3.4. As an exception to the stated requirement

for transaction atomicity, the EIS need not perform automatic recovery for distributed transactions. However, in this event, the EIS must assist in the correct recovery; for example, the EIS must retain all information relevant for recovery and must protect the objects affected by such a transaction against further access until the recovery is complete.

3.8.1.2. The EIS must make the location of the data it manages invisible to a client (except for data administration functions). For example, clients must be able to find and access EIS-managed information regardless of location and as if it were not distributed; client-visible identifications of data objects may not be location dependent.

3.8.1.3. The EIS must provide the ability for authorized users to copy/move data objects among data repositories managed by the EIS.

3.8.1.4. A data repository will either contain a complete physical representation of a data object, or not contain the data object at all. That is, a data repository will not contain a partial representation of a data object. (Note that components of a configuration may be distributed if they are included in the configuration by reference rather than by instantiation.)

3.8.1.5. The EIS must be able to synchronize engineering operations that are performed against redundantly stored copies of a design data object, and they must be able to propagate updates against redundantly stored objects.

3.8.2. Access Request Mapping

3.8.2.1. The EIS must be able to make use of external data management facilities for storing the data it manages, such as commercial DBMS and file systems. In particular, the EIS must support the use of data management components differing in interface languages, data models, and access capabilities, within a single EIS. The EIS must make these differences invisible to EIS clients.

3.8.2.2. The EIS must perform mapping of requests for accessing the data it manages into a strategy of access requests against the external data management facilities.

3.8.2.3. Where possible, the EIS must provide a similar service for data exchange requests against data in attached data repositories. The EIS must be able to make the location of this data invisible to the client at his choosing.

3.8.2.4. The mapping service must be capable of handling the requirements for distribution of EIS-managed data, including the redundant storage of data objects in physically distributed data repositories.

3.8.2.5. The EIS should not assume responsibility for providing the maintenance operations on the physical data repositories, nor for interfacing to

the respective maintenance utility programs. In particular, the requirements for invisibility of distribution and heterogeneity do not extend to these functions.

3.8.3. Data Model and Schema

3.8.3.1. The EIS must provide a data model and supporting processing for describing the objects managed by the EIS and the operations permissible against these objects (such a description is called a **schema**), and the mapping between this schema and those of the external data management facilities that contain the actual data occurrences. Examples of data that must be in the schema include management and control data (e.g., object state information, audit trail data, design history data) and engineering data.

3.8.3.2. In addition, the data model must provide for the description of objects accessible in attached data repositories.

3.8.3.3. The EIS must provide a **common EIS database schema** that describes all EIS-managed data classes and operations that need to be modeled uniformly across installations in order to assure the portability of EIS programs, integrated tools, and rule processing programs. The database schema used by a specific organization is a superset of the common EIS database schema.

3.8.3.4. All schema information is considered EIS administrative data in the sense that it must be recorded as EIS-managed data, subject to access, version, configuration and change control, and accessible by clients in the same fashion as other administrative data of the EIS.

3.8.3.5. The data model must support the concept of classes of data objects, references to data objects, properties of data objects, constraints, and operations against data objects.

3.8.3.6. There must be provisions for associating classes or instances of object references, properties, constraints, and operations with a class.

3.8.3.7. There must be support for the concepts of subclasses and instances.

3.8.3.8. There must be support for the concept of aggregating classes or instances of data objects, object references (pointers, in the programming language sense), properties and operations. At least the following methods must be supported:

(a) records
(b) sets
(c) arrays

3.8.3.9. Support for object references must include the concepts of transitive, nested and inverse relationships.

3.8.3.10. The model must support the description of at least the following classes of properties:

(a) enumeration
(b) boolean
(c) integer
(d) noninteger numeric, including various precision levels (of floating point numbers)
(e) character string of fixed and variable length
(f) variable-length uninterpreted data with no EIS-imposed limitation on the length

3.8.3.11. The model must support the description of at least the following classes of constraints:

(a) referential integrity
(b) type overlap
(c) uniqueness
(d) constraints on individual properties, including enumeration and ranges of values
(e) conditions that determine whether a relationship may exist and that may reference the values of other properties and the existence of other relationships (of the type being defined)

3.8.4. Data Management Support

3.8.4.1. The EIS must support execution of the following operations against EIS-managed data:

(a) identification of data objects, and their associations as defined in the schema, including nested and inverse relationships
(b) retrieval and update of identified data objects, and (selectively) their identified associations
(c) creation, replacement, and deletion of identified data objects
(d) arithmetic, logic and string expressions
(e) transitive closure for transitive relationships
(f) operations defined in the schema

3.8.4.2. The EIS must support the execution of comparison operators against EIS managed data, including:

(a) comparison operators for all property and relationship classes supported by the data model, except uninterpreted data
(b) comparison of identity of data objects and classes
(c) comparison operators for all classes of aggregates supported by the data model
(d) quantifiers

3.8.4.3. The EIS must support the selection of EIS-managed data objects through query expressions composed of valid combinations of operations and comparison operators.

3.8.4.4. The EIS must support the validation of all constraints implied by the data model (e.g., type inheritance) or described in the schema (e.g., referential integrity) for the data it manages.

3.8.4.5. The EIS must extend the above capabilities to the execution of operations against attached data repositories where possible.

3.9. Engineering Information Model

3.9.1. EIS Engineering Information Model

3.9.1.1. The EIS must provide a model of the classes of engineering information that are needed to accurately describe the semantics of the information in the engineering environment in which the EIS operates (listed in subsequent requirements). The EIS Engineering Information Model need not be used to actually represent engineering data; this is the purpose of the common exchange format. Rather, it must provide a definition of all information classes and modeling rules needed as the basis for formulating a conceptual framework for information exchange.

3.9.1.2. The EIS Engineering Information Model must describe all types of information needed to specify schematics, behavior (including timing), layout, and test inputs and results for VHSIC integrated circuit designs.

3.9.1.3. In addition, the EIS Engineering Information Model must provide a model of the classes of administrative data for which a uniform representation across different EISs is desirable (e.g., audit trail data).

3.9.1.4. The EIS must provide a well-defined modeling method, and a language for constructing, changing and extending this model.

3.9.1.5. The EIS Engineering Information Model must describe all required classes of information unambiguously. The specification of semantics must be precise and understandable. The information classes together with prescribed modeling rules must ensure that a given combination of facts can be modeled in exactly one way (i.e., such that there cannot be redundant descriptions of the actual engineering information within the context of the model).

3.9.1.6. The Engineering Information Model must capture all information in the common EIS database schema and must use consistent terminology.

3.9.2. Processing Support Requirements

3.9.2.1. The EIS Engineering Information Model must be recorded electronically. The EIS must provide facilities for clients to access the information model. Specifically, the EIS must provide the following capabilities:

(a) selecting a definition by its name or by its identification.
(b) displaying, adding and deleting definitions interactively and non-interactively, optionally (for displaying) in a standard sort order (e.g., alphabetically).
(c) retrieving, adding and deleting definitions from an interactive or non-interactive program.
(d) browsing definitions by class, or by cross references with other model definitions (e.g., finding the definitions referencing the current one)
(e) editing definitions interactively.

3.9.2.2. The EIS must be able to validate the constraints implied by the modeling method. There is no requirement to validate the constraints described by the Engineering Information Model.

3.9.2.3. The Engineering Information Model must be extensible by a user organization in a prescribed fashion that permits identification of such extensions and of information whose interpretation relies on such extensions. The method of identifying the extensions must not rely on the extension itself.

3.9.2.4. *The method of extension must ensure that user extensions will not* conflict with future extensions of the Engineering Information Model.

3.9.2.5. All programs provided by the EIS that operate upon the Engineering Information Model data must be able to deal with user extensions.

3.9.3. Method and Language Requirements

3.9.3.1. The EIS Engineering Information Model must be developed by using a modeling method and language that supports abstract data types. Specifically, the modeling method must meet all requirements stated for the EIS data model in Section 3.8 and in the following additional requirements.

3.9.3.2. The language must provide the ability to describe queries, including parameterized queries against the information described by the Engineering Information Model.

3.9.3.3. The modeling method must be able to support the concept of a parameterized class, that is, a class whose definition is dependent upon one or several parameters, and where the possible range of classes (that instantiate the parameterized class) is obtained by iterating through all valid combinations of the parameters. It is intended that the modeling method not require the instantiation of these classes in the Engineering Information Model in order for them to be used. An example of applying this capability is the modeling of

NAND-gates, implemented in various technologies. The generic package in Ada provides a similar capability.

3.9.3.4. The modeling method must support the definition of aggregates as being composed of primitives and existing aggregates, where methods of composition include aggregating operations such as queries, and rule-based specifications. For example, it must be possible to specify a relationship as a parameterized query.

3.9.3.5. It is desirable that the modeling method support the concept of aggregation rules analogously to type inheritance rules, that is, rules that imply properties of aggregates and constraints that properties of aggregates must meet.

3.9.3.6. The modeling method must support the concept of imprecise values for properties, such as ranges, means with standard deviation, etc.

3.9.3.7. The modeling method must support the description of constraints. Constraints are assertions about the properties and relationships associated with the instances of a given class or a given collection of objects. It must be possible to treat constraints as objects in their own right (including classification, etc). Where possible, the language used for expressing constraints must be consistent with the language used for describing queries.

3.9.3.8. It is not a requirement that all constraints be formally expressible in the modeling language. For example, explanatory text that describes a constraint, or reference to an algorithm that enforces the constraint is permissible. However, for each constraint that **can** be fully expressed in the modeling language, the Engineering Information Model must provide a formal description.

3.9.3.9. The modeling method must provide all primitive classes and aggregation methods needed to produce the EIS Engineering Information Model, and must allow for the definition of new primitive classes and aggregation methods (i.e., the modeling method itself must be extensible in this sense).

3.10. Rule Processing

3.10.1. Rule processing must be supported by programs that implement all required management and control and other rule-based capabilities.

3.10.2. There must be an interface specification for every situation in which rule processing is necessary that allows programs to invoke appropriate rule processing programs and pass parameters to them.

3.10.3. Rule processing may be implemented via object programs in the short term. However, a user organization must be able to influence the execution of

the programs through EIS administrative data. For example, there should be parameters for activating/deactivating rules and specifying actions.

3.10.4. The EIS must be able to invoke the rule processing services in a heterogeneous, distributed environment. The services must fulfill tool availability requirements stated in Section 3.1.

3.11. Control Points

3.11.1. Identification of Control Points

3.11.1.1. The EIS must allow a user organization to associate a control point with each type of event that must be recognized for management and control purposes. Such events include, for example:

(a) client login/logout
(b) program invocation/termination
(c) data access
(d) data exchange

3.11.1.2. The EIS must recognize automatically the occurrence of those control points that are associated with the initiation and completion of operations performed by the EIS or with errors and exceptions that the EIS is expected to handle.

3.11.1.3. In addition, the EIS must provide means for clients to inform it of the occurrence of a control point and must provide interface specifications serving that purpose.

3.11.2. Invocation of Control Points

3.11.2.1. When a control point occurs, the EIS must be able to invoke rule processing services consistent with the rule processing requirements.

3.11.2.2. At each control point, the presence of controls must be optional.

3.12. User Interface

3.12.1. User Interface Guidelines

3.12.1.1. User interface guidelines must be developed that will provide the basis for designing and preparing the specifications for the EIS user interface, and for extending them to aspects of interactive tool interfaces. These guidelines must cover:

(a) screen layout, including the use of separately controllable subsets of a display (e.g., windows)

(b) methods for interacting with concurrently executing processes
(c) use and structure of icons and graphic symbols
(d) commands, menus, and prompting in textual and graphic format
(e) use of color
(f) method of invoking, suspending, and exiting the EIS user interface processor
(g) method of invoking help, including context sensitive help
(h) consistent semantics for commands and function keys
(i) a consistent system for the structure and meaning of error messages

3.12.1.2. The user interface guidelines must specify a uniform approach to common functions across all levels of the interface facilities and terminal types, for example:

(a) textual objects must be available on graphic terminals.
(b) keyboard commands must be available on terminals that provide pointing devices.

3.12.1.3. The user interface must be adaptable to the needs of particular end users or groups of end users. The guidelines must cover the necessary features such as:

(a) operating in multiple modes to support a range of users from experts to novices, including providing terse and verbose modes.
(b) allowing user-specific defaults based on stored user profiles.

3.12.2. User Interface Processor

3.12.2.1. An EIS user interface processor must be developed that supports all required EIS interface facilities, including:

(a) invoking tools and EIS services.
(b) interacting with EIS services, including performing all EIS administrative functions.

3.12.2.2. In addition, the EIS user interface processor must support:

(a) requesting and displaying context-sensitive help messages.
(b) escaping to the host operating system.

3.12.2.3. The interface processor must be developed in accordance with the user interface guidelines.

3.12.2.4. The user interface processor must hide from the user the heterogeneity of the system. The user must be presented with a consistent interface irrespective of the host to which the user is connected.

3.12.2.5. A program interface must be specified that permits EIS programs to access the user interface processor.

3.12.3. Object Editors

No short-term requirements.

3.12.4. Description Driven Characteristics

3.12.4.1. The syntax and display formats used by the user interface processor in the tool invocation dialog must be driven by the descriptions of the objects that can be invoked by, and are registered with, the EIS.

3.12.4.2. The attributes of alphanumeric and graphic symbols used by the user interface processor must be description driven.

3.12.4.3. The relative placement and scale of alphanumeric and graphic symbols that describe the same object, or objects within the same aggregate, must be description driven.

3.12.4.4. The user interface processor must be extensible in the sense that registering a new tool or service with the EIS and providing the required information about that tool or service will be sufficient to extend the user interface to support its invocation.

3.13. EIS Administration

This section presents requirements for EIS administrative data objects, including specific lists of information that must be included in the description of some of those objects. It also presents requirements for specific EIS administrative functions.

3.13.1. Registration, Maintenance and Use of Administrative Data

3.13.1.1. The EIS must support the maintenance of administrative information as EIS-managed data, as specified in Section 3.4. This data includes all information required to operate the EIS and execute the management and control policies. The following are examples of administrative data objects:

(a) authorized users
(b) tools
(c) available host environments
(d) networks
(e) engineering and management operations
(f) control points
(g) audit trail and design history
(h) EIS database schema
(i) rule processing information

3.13.1.2. All administrative data objects must be controlled by the EIS as specified in Section 3.3. These controls include, for example: configuration management, version control, change control, and access control.

3.13.1.3. If the EIS replicates any of this data (e.g., caches it on a host to improve availability or performance), it must support automatic revision of the replicated data when they are affected by a change.

3.13.2. Administrative Data: Authorized Users

3.13.2.1. Specific information about authorized users must at least include:

(a) user identification
(b) descriptive data, such as name, title, and function
(c) data needed for user authentication (e.g., a password)
(d) user authorizations
(e) level of expertise, interactive style, etc.

3.13.2.2. An authorized client must be permitted to change authentication data.

3.13.3. Administrative Data: Tools

3.13.3.1. It must be possible to include at least the following specific information about each tool:

(a) tool name
(b) textual description of tool's function
(c) description of the required host support environment
(d) description of the user interface device classes that are supported
(e) information required to initiate the tool execution on all host environments supporting it
(f) tool parameter definitions as needed to validate tool parameter instantiations (see Section 3.1), including data types, formats, and representations of both inputs and outputs
(g) preconditions for tool execution and any postconditions or types of consistency among objects which the tool ensures
(h) data regarding valid modes of invocation for the tool
(i) data needed to authenticate the tools
(j) tool authorizations

3.13.4. Administrative Data: Host Environments

3.13.4.1. Specific information about each available host environment must at least include:

(a) unique identifier
(b) information needed to access its EIS host interface
(c) identification of which aspects of the EIS host interface are supported

(d) information about the system services that are globally accessible at the host
(e) description of the host operating environment for use during tool initiation (tool builders should be able to rely upon this information to program self-configuring tools)

3.13.5. Administrative Data: Networks

3.13.5.1. In addition to identification of the specific tools and hosts available, information that must be maintained about the EIS **internal network** (that is, the network connecting hosts within an EIS) at least includes:

(a) information required by the communication systems and/or the EIS to correctly transfer data
(b) access control rules governing access to the network
(c) audit trail and other administrative rules governing use of the network
(d) information needed to ensure the security of the data during transfer

3.13.5.2. A user support organization must be provided with capabilities that permit it to substitute names or identifications of its own choice for destination addresses employed by the communication system.

3.13.6. Administrative Data: Operations

3.13.6.1. Information that must be maintained about engineering and management operations includes, for example:

(a) whether the operation is manual or automated
(b) which tools can be used to perform the operation
(c) a list of conflicting operations

3.13.7. Governing EIS System Policy

3.13.7.1. Authorized clients must be able to specify and change, in interactive or noninteractive mode, the rules and/or state information that define at least the following system policies, as well as their effectivity:

(a) data exchange within an EIS
(b) configuration management
(c) data dependency tracking
(d) change control
(e) versioning
(f) access controls
(g) extent of the system audit trail
(h) automated tool networks
(i) concurrent access to engineering data

3.13.7.2. The EIS must ensure that policy changes take effect without uncontrolled impact upon current operations that are affected by the policies.

3.13.8. Accounting

No short-term requirements.

3.13.9. Performance and Operations Monitoring

No short-term requirements.

3.13.10. System Maintenance

No short-term requirements.

3.14. Programmatic Interfaces

3.14.1. Tool Interface

3.14.1.1. The EIS must specify Ada interface packages for all types of service requests that are supported through the EIS. Examples include: data exchange, tool invocation, methodology execution.

3.14.1.2. The interface packages must be logical extensions of CAIS. That is, CAIS specifications must be used where they exist; otherwise, the interface packages must remain compatible with general CAIS requirements.

3.14.1.3. The EIS must specify a host language interface for accessing data objects stored in integrated and attached data repositories. The host language interface must be consistent with the requirements stated in Section 3.8 for access to data. In particular, the host language interface must be:

(a) consistent with the data model underlying the EIS database schema.
(b) able to support all specified data access requirements.
(c) consistent with the distributed data management requirements (e.g., support location transparency).

3.14.2. Host/Workstation Interface

3.14.2.1. The EIS must specify a host/workstation interface that meets the host/workstation interface requirements in Section 3.1, and that includes specification of program-callable interfaces as well as the necessary high-level protocols (e.g., for exchanging requests, status information, and design data).

3.14.3. Interface to User Interface Processor

3.14.3.1. The EIS must provide an interface specification that will allow programs, including the EIS itself, to utilize the services of the EIS user interface processor, and that is consistent with the user interface requirements in Section 3.12.

3.14.4. Interface to Rule Processing

3.14.4.1. The EIS must provide an interface specification that will allow programs, including the EIS itself, to utilize rule processing services, and that is consistent with the rule processing requirements in Section 3.10.

3.14.5. Interface to System Services

3.14.5.1. The EIS must provide specifications for the back-end interfaces to operating system services, database system services, and other external software services used.

3.14.6. Programming Language Support

3.14.6.1. All programmatic interfaces must be supported for invocation by Ada programs.

3.14.6.2. The EIS must provide interface specifications for invocation by C programs that are semantically equivalent to those supported for invocation by Ada programs, and must support invocation via these interfaces.

3.15. Design and Implementation Requirements

3.15.1. Portability Requirements

3.15.1.1. The EIS software must be portable among run-time environments. Specifically, the EIS must use the specified EIS programmatic interfaces for utilizing EIS-internal and external software services.

3.15.2. Adaptability Requirements

3.15.2.1. The EIS may not assume that all EIS system configurations are identical.

3.15.2.2. The EIS must support a heterogeneous environment. That is, the EIS may not assume that in an EIS configuration each system service is provided by one unique component.

3.15.2.3. The EIS must assume that services and resources in the environment are owned, and therefore controlled, by different organizations.

3.15.2.4. The EIS must not assume that all data exchange is routed through a central facility or uses an integrated data repository as the means for exchanging data.

3.15.2.5. The EIS facilities may not be limited to accepting, transporting, storing and delivering only information represented in a common EIS format.

Specifically, the EIS may not assume that all data in the integrated and attached data repositories are in a common format.

3.15.2.6. The EIS must not limit an EIS user organization in its choice of: engineering methods and tools; management policies, methods, and tools; security and access control policies and tools.

3.15.2.7. The design and implementation of the distributed operating system and data management facilities must be structured such that they can be bypassed in favor of, or replaced by, external software services, when such services become available.

3.15.3. Installation and Maintenance Requirements

No short-term requirements.

3.15.4. Extensibility Requirements

No short-term requirements.

3.15.5. Integrity/Security Requirements

No short-term requirements.

4. Extended Short-term Requirements

4.1. Tool/Workstation Integration

4.2. Data Exchange

4.3. Engineering Management and Control

No extended short-term requirements for the Sections 4.1, 4.2, and 4.3.

4.4. Information Management

4.4.1. Whereas the EIS must provide, in the short term, a common database schema that is **compatible with** the Engineering Information Model (see Section 3.9), it is an extended short-term requirement that the EIS support the Engineering Information Model itself as the common EIS database schema. Specifically, the EIS must make the decomposition of design data objects and their relationships to other design data objects visible to clients according to that model. This includes the requirements for supporting update and data extraction operations against design data objects, for navigating within a design data object (according to its decomposition) and among design data objects according to their relationships, and for triggering constraint processing. The EIS is not required to support execution of the full scope of the query and constraint specification language provided with this model.

4.4.2. The EIS must prevent programs from inadvertently accessing information in the Engineering Information Model whose interpretation has been changed by extensions.

4.4.3. The EIS must provide methods for automatically converting between a common exchange format and information stored according to the Engineering Information Model, except where information is represented as uninterpreted strings.

4.4.4. The EIS must support browsing of information in a design data object that is stored in a common exchange format or according to the common EIS database schema (except for uninterpreted strings).

4.4.5. The EIS must support the selection, insertion, and replacement of information in a design data object stored in a common exchange format or according to the common EIS database schema, according to control input provided by a client. However, the EIS is not required to evaluate uninterpreted data for that purpose.

4.4.6. The EIS must support the use of this capability for maintaining design descriptions as specified in Section 3.4.2.

4.5. External System Interfaces

4.6. Object Management System

4.7. Distributed Operating System Facilities

4.8. Distributed Data Management Facilities

No extended short-term requirements for Sections 4.5., 4.6, 4.7, and 4.8.

4.9. Engineering Information Model

See Section 4.4, above, for related extended short-term requirements.

4.10. Rule Processing

4.10.1. All rule-based capabilities required by the EIS must be provided by a rule processor, which can be invoked through programs that use the specified standard interfaces.

4.10.2. The rule processor must support the execution of rules specified by a rule specification language.

4.10.3. The EIS must support facilities for adding, deleting, and modifying rules.

4.10.4. The rule specification language must support the concept of system-supplied variables, such as date, and must support evaluation of expressions, condition testing, and the triggering of actions.

4.10.5. The rule specification language must allow for the specification of actions, including sending messages, changing global and object-related management and control information, and invoking programs.

4.10.6. The rule specification language must support the concept of variables and parameters.

4.10.7. The rule specification language must permit use of any type of object as a variable or parameter and must allow for the specification of parameterized queries containing update operations against EIS-managed data.

4.10.8. The EIS, in combination with the rule processor, must be able to support the concept of parameterized messages and programs, and must be able to supply the parameter instantiations automatically.

4.11. Control Points

No extended short-term requirements.

4.12. User Interface

4.12.1. The user interface guidelines must be extended to address the interaction between users and interactive tools.

4.12.2. The guidelines must provide a consistent approach for editing:

(a) textual objects
(b) 1-D, 2-D, and 3-D graphic objects relevant to electronic design
(c) electronic design objects

4.12.3. An interface must be specified that permits integrated tools to access a user interface processor implementing these guidelines.

4.12.4. It is desirable that these guidelines support the concept of object editors as defined in the long-term requirements.

4.13. EIS Administration

4.14. Programmatic Interfaces

4.15. Design and Implementation Requirements

No extended short-term requirements for Sections 4.13, 4.14, and 4.15.

5. Long-term Requirements

5.1. Tool/Workstation Integration

5.1.1. Tool Initiation

5.1.1.1. The EIS must provide for the invocation of integrated and designated attached interactive tools, locally and remotely, including in a host environment which is different from the environment of the client, and at any terminal (subject to hardware limitations).

5.1.1.2. The EIS must provide the ability to execute integrated interactive tools in a non-interactive mode.

5.1.2. Parameter Handling

No additional long-term requirements.

5.1.3. Execution Control

5.1.3.1. The EIS must allow the execution of a tool to be suspended and restarted, both at client direction and in the case of client termination/reconnection.

5.1.3.2. The EIS must provide a capability which allows previous tool executions to be undone and redone.

5.1.4. Data Setup and Disposition

No additional long-term requirements.

5.1.5. Automated Program Networks

5.1.5.1. The EIS must be able to engage in a user dialog during the execution of an automated program network, for example, to prompt for additional tool parameters, control input, and authorizations. This dialog must occur automatically when such input is needed. Automated program networks must allow for specification of such dialog requirements.

5.1.5.2. The EIS must permit the inclusion of interactive programs into automated program networks.

5.1.6. Workstation/Host Interface

No additional long-term requirements.

5.1.7. Version/Configuration Management for Tools

5.1.7.1. The EIS must be able to identify incompatibilities in data input/output and other operational characteristics that may exist among different versions of the same tool configuration.

5.1.8. Tool Back-up

5.1.8.1. The EIS must be able to back-up tool configurations when new tool versions are installed, and restore them when the need for using an old tool version arises.

5.2. Data Exchange

5.2.1. General Data Exchange Requirements

No additional long-term requirements.

5.2.2. Data Exchange Formats

No additional long-term requirements.

5.2.3. Data Exchange Support Requirements

5.2.3.1. There must be definitions of common standard deliverables that can be used as reference guidelines for the types of information that should be provided as the result of a specified design activity or by a tool set. The definitions must employ the terminology of the EIS Engineering Information Model. Examples of such standard deliverables are: the definition of the information that is considered to constitute a complete software/firmware program, schematic, netlist, or layout for a given technology, and test information for a given validation requirement.

5.2.3.2. The EIS environment must contain programs for translating between the common exchange format and formats that are employed by widely-used tool sets.

5.2.3.3. The EIS environment must contain building blocks and methods that can be used to construct and maintain such translators more cost-effectively.

5.2.3.4. The EIS must be able to decide automatically which interactive and noninteractive programs need to be invoked for translating among different methods of design representation and hardware-dependent encoding of data, and for extracting a subset of the information in a data object. The decision must be based upon the representation, formatting, and data requirements of the source and target data object type or tool I/O stream.

5.2.3.5. The EIS must provide programs that can perform the following functions for design data objects stored in a common format:

(a) automatic extraction of information from the data object and the data objects referenced by it, based on specified data requirements
(b) automatic calculation of derived data, provided that a derivation method for the respective data type has been specified

5.2.4. Exchange of Management Data

No additional long-term requirements.

5.3. Engineering Management and Control

5.3.1. Object Registration

No additional long-term requirements.

5.3.2. Configuration Management

5.3.2.1. A user organization must be able to define a configuration by using rules or parameterized database queries and by associating configurations with an object class, including the specification of a default configuration.

5.3.2.2. Clients must be able to define and use *ad hoc* configurations that may or may not be subsequently stored.

5.3.2.3. As part of its support for partial configurations, the EIS must be able to report to specified clients when an operation fails because a partial configuration is used, and to record the fact in the audit trail.

5.3.3. Dependency Tracking

No additional long-term requirements.

5.3.4. Change Control

5.3.4.1. The EIS must be able to recognize that an object waiting for authorization has been authorized (e.g., an operation).

5.3.4.2. It must be possible to specify the retroactivity of changes to design data objects with respect to existing configurations of which they are member, at the time the change is made, and as part of the management and control information associated with a configuration.

5.3.4.3. The EIS must be capable of supporting the following additional change control actions:

(a) changing the methodology state.
(b) including in messages information from the objects considered in the decision as well as information from the audit trail.
(c) invoking tools and system services, and including information from the object, dependent objects and objects in the same configurations as well as audit trail data in the invocation parameters.

5.3.5. Version Control

No additional long-term requirements.

5.3.6. Security and Access Control

5.3.6.1. Security and access control capabilities of the EIS must be compatible with the long-term design and implementation requirements of the EIS.

5.3.7. Audit Trail

No additional long-term requirements.

5.3.8. Design History

5.3.8.1. The EIS must be able to construct a complete design history of a design data object or configuration and provide it to a requesting client.

5.3.8.2. The EIS must be able to present a design history or a design configuration as a directed graph reflecting:

(a) design decomposition
(b) versions
(c) dependencies
(d) alternatives
(e) design operations
(f) time ordering

5.3.8.3. The EIS must be able to include in the design history the same types of information that are captured in an audit trail (see Section 3.3.7), at the option of the client.

5.3.8.4. The EIS must permit a client to query the design history (i.e., select a partial history) according to specifications of the client.

5.3.9. Methodology Support

5.3.9.1. A user organization must be able to define methodologies that cover the complete design life cycle, including specification of the engineering tasks

to which they may be applied, and the decomposition into series of steps that in turn are either engineering operations or methodologies. The decomposition must support:

(a) alternatives and decision (e.g., approval or check-off) points
(b) the ordering of steps in a directed graph
(c) the specification of default, desired and mandatory events

5.3.9.2. A user organization must be able to associate management and control policies with a methodology, such as design libraries to be used, and rule sets to be employed at the control points.

5.3.9.3. The EIS must allow different methodologies to reference the same sequence of steps.

5.3.9.4. The EIS must support loops in methodology definitions.

5.3.9.5. The EIS must support methodologies that are capable of sequencing tool execution and other actions based upon arbitrary tests involving multiple objects, including the following conditions or combinations thereof:

(a) validity checks, such as a determination that references are to existing objects
(b) consistency checks or status that involve multiple representations of an object
(c) dependencies between objects
(d) previous results of tool executions in combination with records of inputs and tool versions
(e) the current state of any methodology-defined parameters

5.3.9.6. An authorized client must be able to assign a methodology to a planned or current engineering project. However, the system must not require that each project or task be assigned a methodology.

5.3.9.7. The EIS must support changes in methodology states through control points.

5.3.9.8. The EIS must be able to enforce the methodology where the steps represent the execution of requests via the EIS.

5.3.9.9. An authorized client must be permitted to deviate from the methodology (e.g., must be permitted to make use of tools in an *ad hoc* fashion).

5.3.9.10. A client may have more than one methodology (or the same methodology over more than one task) active at one time. However, the EIS must be able to determine unambiguously which methodologies apply to a given task.

5.3.9.11. The EIS must support methodologies that allow an authorized client to circumvent or short-circuit steps of the methodology. The EIS also must be capable of requiring such a client to provide a reason or explanation for this action and record this information in the audit trail.

5.3.9.12. The EIS must be able to track, as part of the design history requirement, all deviations from a methodology or a default, choices among alternative methodologies or operations, or actual methodology where none was prescribed.

5.3.9.13. The EIS must provide the capability to step the client through the methodology in a tutorial fashion. Part of this capability would include the provision of a context-sensitive help capability.

5.3.9.14. The EIS must provide a capability for reviewing the steps that previously had been executed as a part of the methodology.

5.3.9.15. The EIS must support a methodology that directs the update of EIS-managed data based upon the results of tool execution and that maintains the dependency of the results on input files, run parameters, and tool versions.

5.3.9.16. The EIS must support a methodology that requires automatic rerun of previously executed steps for new versions of the same design. This would include the capability to use the same or updated versions of control inputs, design libraries, and/or tools.

5.3.9.17. The EIS environment must contain tools to assist the user organizations in choosing, creating, and maintaining methodologies.

5.3.9.18. The EIS environment must contain tools to determine whether a given design history satisfies a given methodology.

5.3.10. Authorization and Approval

5.3.10.1. The EIS must be able to include and sufficiently protect data to capture client signatures. A client signature must be unique.

5.3.10.2. It must be possible for a number of signature roles to be associated with the same object, each indicating a different type or level of approval or authorization.

5.3.10.3. It must be possible for a number of clients to be authorized to affix signatures for the same signature role.

5.3.10.4. The EIS must ensure that signatures may not be forged; exactly one client must be authorized to write a given signature.

5.3.10.5. Authorized clients must be able to determine whether or not a signature is affixed for a certain signature role and object.

5.3.10.6. Authorized clients must be able to determine the identity of any client whose signature is affixed for any particular signature role and object.

5.3.10.7. It must be possible to affix signatures interactively and non-interactively.

5.3.10.8. The EIS must be able to prompt clients for their signatures when the need for their signatures arises.

5.3.10.9. A client must be able to revoke his signature as long as the actions depending upon the signature have not been started.

5.4. Information Management

5.4.1. Management of Design Data Objects

5.4.2. Administrative and Descriptive Information

See Section 5.9 for additional long-term requirements in the above areas.

5.4.3. Automatic Versioning of Objects

No additional long-term requirements.

5.4.4. Support for Concurrent Access to Design Data Objects

5.4.4.1. The EIS must be able to queue specified classes of engineering operations (i.e., to hold a request that conflicts with an ongoing operation and to consider that request when the ongoing operation completes). Authorized clients must be able to specify priority of de-queuing and to manage (i.e., query and change) queues. The EIS must be able to detect and resolve deadlocks. It must be possible to associate queuing and dequeuing with control points.

5.4.5. Interim Modifications to Design Data Objects

5.4.5.1. The EIS must be able to keep track of interim modifications made by clients performing an engineering operation.

5.4.5.2. An authorized client must be able to specify when interim modifications should be tracked.

5.4.5.3. The EIS must be able to incorporate into a data object interim modifications upon request by an authorized user or automatically (e.g,. when the system determines that an engineering operation has terminated abnormally).

5.4.5.4. An authorized client must be able to read interim modifications.

5.4.5.5. Upon normal completion of an engineering operation, the EIS must be able to recognize that the new version of the object supersedes the interim modifications.

5.4.6. Backup and Recovery of Objects

5.4.6.1. In addition to those backups taken for purposes of system recovery, the EIS must be able to create and delete back-up copies, according to an *ad hoc* request or default specifications, for purposes of check-pointing the design.

5.4.6.2. The EIS must be able to associate with back-up copies all relevant administrative information, including, for example:

(a) indication that this is a back-up copy
(b) the time the back-up was made
(c) identification (including version number) of the particular object being backed up
(d) identifier of the client/user initiating the backup

5.4.6.3. The EIS must be able to accommodate routine back-ups of data objects and configurations, with a frequency determined by the user organization.

5.4.6.4. Authorized clients must be able to read back-up copies of data objects and configurations and to install them as primary copies.

5.4.6.5. If a back-up copy is installed as a primary version of an existing object, the system must perform change control.

5.4.6.6. If a back-up copy of an object is installed as the primary version of a new object, the system must create the appropriate dependency information.

5.4.7. Archiving Objects

5.4.7.1. The EIS must be able to archive design data objects and configurations according to an *ad hoc* request or default specifications.

5.4.7.2. When a design data object or configuration is archived, the EIS must be able to include all relevant administrative data.

5.4.7.3. The EIS must support archival to off-line, long-term storage media, including tape.

5.4.7.4. The EIS must support requests for archived data, but access to that data may require manual intervention (such as mounting a tape).

5.5. External System Interfaces

5.5.1. Invocation of Tools across EIS boundaries

5.5.1.1. The capabilities provided by the EIS for the invocation of tools and services across EIS boundaries must be similar to and consistent with those provided within a single EIS.

5.5.2. Data Exchange among Different EISs

5.5.2.1. The capabilities and mechanisms for exchanging data among different EISs must be similar to and consistent with those supported for exchanging data within a single EIS.

5.5.2.2. The EIS must permit a user organization to define requirements for management and control data that must be present in data received from an external system, and for actions to be taken if the required data is missing.

5.5.3. Data Exchange with non-EIS Systems

5.5.3.1. The EIS must provide a mechanism for mass-transfer of information into and out of the EIS for the purpose of exchanging information with non-EIS systems.

5.5.3.2. The EIS must provide a mechanism for accepting and applying updates that were received from non-EIS systems, and for supplying interim modifications to non-EIS systems.

5.5.4. Management and Control of External System Interfaces

5.5.4.1. The EIS must provide a mechanism for authenticating clients from another EIS prior to making any of its services available to those users (this is called remote login). The EIS must verify that all management and control information required by the local user organization has been received, and must use this information to determine whether the login request will be accepted. The EIS must be able to invoke programs provided by the local user organization for validation purposes on this occasion.

5.5.4.2. The EIS must be able to perform the login for its clients at a remote EIS automatically. It must provide automatically all management and control information that is required for that purpose to the remote EIS. It must handle all interactions with the remote system that are required to process and approve or deny the login request transparently to the client. The EIS also must support the requirements of a user organization for not accepting authentications of a remote system and insisting on reauthentication of the client.

5.5.4.3. The EIS must subject all operations performed by a local or remote client via the external system interface to the locally defined management and

control rules. The EIS must be able to restrict client capabilities based on the fact that the client is remote.

5.5.4.4. The EIS must be able to perform change control for the copy of a transmitted object or configuration at the receiving EIS, at the discretion of the sending organization and subject to approval by the receiving organization. The EIS must be able to determine/approve the change control policies automatically as a function of management and control information associated with the object to be transmitted, and the origin and destination of the transmission.

5.5.4.5. The EIS must be able to support the following change control actions in addition to the general change control requirements that apply to data exchange within a single EIS:

(a) notify clients in the local and remote EIS.
(b) require transmission of the modified object or configuration or of the accumulated interim modifications, or trigger automatic transmission after the changed object/configuration has reached a given release status.
(c) require acknowledgement of notifications and transmissions.

5.5.4.6. The EIS must be able to identify in the local audit trail any operations performed by clients from a remote EIS, and any operations on objects that have been received from a remote EIS.

5.5.4.7. The EIS must provide a capability to exchange this audit trail data with the remote EIS according to *ad hoc* and default specifications.

5.6. Object Management System

No additional long-term requirements.

5.7. Distributed Operating System Facilities

5.7.1. Queuing and Scheduling

No additional long-term requirements.

5.7.2. Interconnection Services

5.7.2.1. The EIS must provide gateways and forwarding services for interconnecting different types of communication services.

5.7.3. Request Mapping

5.7.3.1. The EIS must provide a rule-based method for optimizing plans with respect to use of resources, turn-around time, and other cost and performance measures.

5.7.4. Process Monitoring

No additional long-term requirements.

5.7.5. Error and Exception Handling

No additional long-term requirements.

5.8. Distributed Data Management Facilities

5.8.1. Data Distribution

No additional long-term requirements.

5.8.2. Access Request Mapping

5.8.2.1. The EIS must optimize the access strategy. For example, the EIS must take into account the cost of the access operations and data transfer operations. The EIS must be able to distribute the data manipulations required to execute a request over a computer network.

5.8.3. Data Model and Schema

5.8.4. Data Management Support

See Section 5.9 for additional long-term requirements in the above areas.

5.9. Engineering Information Model

5.9.1. EIS Engineering Information Model

5.9.1.1. The EIS Engineering Information Model must include at least the following classes of information:

(a) administrative information
(b) requirements and requirements derivation
(c) system architecture, components, and interfaces (hardware, software, and firmware)
(d) program descriptions, including microprograms
(e) behavior, including the behavior of software programs
(f) software/hardware configurations

(g) procedural descriptions
(h) test data and test strategies
(i) state diagrams
(j) schematics, block diagrams, stick diagrams, floor plans, layouts
(k) geometries and physical structure
(l) logic elements, components, logic diagrams and logical structure
(m) circuit elements
(n) data dependency graphs, sequence data and concurrency
(o) pipelining
(p) design rules
(q) material properties
(r) power consumption
(s) reliability models
(t) failure modes
(u) queuing models

5.9.2. Processing Support Requirements

5.9.2.1. The EIS must be able to support the Engineering Information Model as its schema, process all operations, constraints and queries expressible in the modeling language, and provide the same processing support for the Engineering Information Model data as for EIS administrative data.

5.9.2.2. The EIS must enforce all constraints described in the model, including the automatic invocation of constraint checking programs referenced in the model.

5.9.2.3. Where possible, the EIS must validate that all user extensions follow the EIS rules for such extensions.

5.9.3. Method and Language Requirements

No additional long-term requirements.

5.10. Rule Processing

5.10.1. There must be standard definitions for frequently-needed policies, e.g., of government-mandated policies such as required by MIL-standards. The EIS environment must include an implementation of these policies.

5.11. Control Points

5.11.1. Identification of Control Points

No additional long-term requirements.

5.11.2. Invocation of Control Points

5.11.2.1. A mechanism must be provided that will validate tool adherence to the EIS control points.

5.12. User Interface

5.12.1. User Interface Guidelines

No additional long-term requirements.

5.12.2. User Interface Processor

5.12.2.1. The EIS user interface processor must provide type-ahead and logging facilities.

5.12.2.2. The EIS user interface processor must support a noninteractive execution mode that is transparent to interactive programs.

5 12.2.3. The EIS user interface processor must provide an interface for object editors to be used for display of objects and for converting screen input into textual or graphical objects, changes to objects, or messages.

5.12.2.4. The EIS user interface processor must allow a program to specify that an object editor is to be used to display or edit an object or part of an object.

5.12.3. Object Editors

5.12.3.1. Object editors must be provided for all classes of objects in the EIS Engineering Information Model.

5.12.3.2. EIS supported object editors must provide facilities for reconstructing partially edited objects after a system failure.

5.12.3.3. A generic object editor is required that can be used for editing all classes of objects, including design and engineering objects, which are managed by the EIS.

5.12.4. Description Driven Characteristics

5.12.4.1. The EIS user interface processor must be able to vary the display characteristics of alphanumeric and graphic symbols based on the current screen and window context (e.g., crowding of objects, available space on the screen, current area of interest, appropriate treatment of hidden lines and screen border areas).

5.12.4.2. The same description driven characteristics that apply to the user interface processor must extend to object editors, and the user interface of

object editors must be extensible in the sense that interfaces and/or formats can be added or altered by adding new descriptions, or altering existing descriptions that are maintained by the system.

5.13. EIS Administration

5.13.1. Registration, Maintenance and Use of Administrative Data

5.13.2. Administrative Data: Authorized Users

No additional long-term requirements.

5.13.3. Administrative Data: Tools

5.13.3.1. It must be possible to include the following information about each tool:

(a) tool history
(b) problem history
(c) known limitations
(d) run-time resource requirements
(e) representative examples of usage in the form of simulated executions

5.13.4. Administrative Data: Host Environments

No additional long-term requirements.

5.13.5. Administrative Data: Networks

5.13.5.1. The EIS must provide the facilities to maintain the information related to its external interfaces. In addition to the information needed for managing the EIS internal network, the information that must be maintained about the EIS **external network** includes:

(a) design management rules governing the release of design information to the destination
(b) audit trail generation rules and other administrative rules governing use of the external interface services

5.13.6. Administrative Data: Operations

No additional long-term requirements.

5.13.7. Governing EIS System Policy

5.13.7.1. Authorized clients must be able to specify and change, in interactive and noninteractive mode, the administrative information that defines the following system policies, as well as their effectivity:

(a) the capability of a particular client to affix a signature to a particular signature role of a data object
(b) storage and incorporation of interim modifications
(c) back-up copy creation and deletion
(d) archiving
(e) methodologies

5.13.8. Accounting

5.13.8.1. The EIS must collect information necessary for cost accounting purposes, including:

(a) identification of client(s) who are using system resources
(b) identification of the resource and type of usage
(c) measure of the system resource usage
(d) account number to be charged for the resource usage
(e) time stamp
(f) information necessary to uniquely relate the resource utilization to a step in the design history

5.13.8.2. The EIS must be capable of interfacing at run-time with cost-accounting packages and to provide the information cited above to them.

5.13.9. Performance and Operations Monitoring

5.13.9.1. The EIS must collect information necessary to judge its current performance as well as its performance history. The following are examples of information that is considered necessary:

(a) wait times for requests and reasons for wait
(b) system resource utilization by EIS services and by services that are used by the EIS to satisfy a client request

5.13.9.2. The EIS must be capable of interfacing at run-time with performance-monitoring packages and to provide the information cited above to them.

5.13.9.3. Authorized clients must be able to monitor ongoing operations and terminate them. They must be able to start up, enable, disable, and shut down operation of individual EIS components.

5.13.10. System Maintenance

5.13.10.1. Authorized clients must have the ability to put new versions of EIS objects or configurations into effect. The EIS must support control points for these actions.

5.13.10.2. The EIS must prevent inconsistent configurations of EIS components from being put into operation. For example, the EIS must prevent EIS utilities or services from executing upon EIS administrative data with which they are inconsistent. The EIS must also prevent the use of outdated versions of cached administrative data in the execution of requests where this is clearly undesirable.

5.14. Programmatic Interfaces

5.14.1. Tool Interface

5.14.1.1. The EIS environment must contain programs for attaching widely-used tool sets to the EIS.

5.14.2. Host/Workstation Interface

5.14.2.1. The EIS environment must contain programs for attaching widely-employed hosts to the EIS.

5.14.3. Interface to User Interface Processor

No additional long-term requirements.

5.14.4. Interface to Rule Processing

No additional long-term requirements.

5.14.5. Interface to System Services

5.14.5.1. The EIS environment must contain programs that map from the EIS interface formats to widely employed system services, including:

(a) mapping from the EIS interfaces for communication among EIS clients to specific communication systems.
(b) mapping from EIS data access and management interfaces to specific data manipulation or data definition languages.

5.14.6. Programming Language Support

5.14.6.1. The EIS must provide semantically equivalent interface specifications for invocation by Ada, C, Fortran, and Common Lisp programs, and must support invocation via these interfaces.

5.15. Design and Implementation Requirements

5.15.1. Portability Requirements

No additional long-term requirements.

5.15.2. Adaptability Requirements

5.15.2.1. The EIS must permit an EIS user organization to employ off-the-shelf products in configuring the required system services to the maximum extent possible.

5.15.3. Installation and Maintenance Requirements

5.15.3.1. The EIS must provide aids for its installation and maintenance by a user support organization.

5.15.3.2. The EIS must include run-time provisions that ensure that it is configured correctly.

5.15.3.3. Clients must be able to query the configuration of the EIS and of system services.

5.15.4. Extensibility Requirements

5.15.4.1. It must be possible to extend all functionality of the EIS to all user-defined extensions of the common database schema, the EIS engineering information model, and the EIS common representation(s).

5.15.5. Integrity/Security Requirements

5.15.5.1. The reliability, availability, and survivability of a system must not be degraded by virtue of using the EIS.

5.15.5.2. It must be possible to operate the EIS in a trusted computer system.

5.15.5.3. The EIS must be able to guarantee the authenticity of all data objects it manages, and during a data transfer, the integrity of all data being transferred.

5.15.5.4. The EIS must provide security and integrity capabilities needed to support the exchange of classified design information among different EISs.

Distribution List for IDA P-1953

Sponsor

Dr. Egbert Maynard (5 copies)
VHSIC Program Office
1211 Fern St., C-112
Arlington, VA 22202
(202) 697-4198

Other

Mr. Harvey Alperin
Naval Surface Weapons Center
Code R45
Silver Spring, MD 20910

Mr. Charles Bosco
U.S. Army Electronic Technical & Service Laboratory (LABCOM)
SLECT-IC
Ft. Monmouth, NJ 07703

Capt. Anthony Gadient
AFWAL/AADE-1
Wright-Patterson AFB, OH 45433-6543
(513) 255-8641

LCDR. Pat Gariano
Room 7W20
Bldg 1
National Center
Washington D.C. 20363-5100

Dr. John Hines
AFWAL/AAD-VPO
Wright-Patterson AFB, OH 45433-6543
(513) 255-3503

Mr. Stanley Wagner
AFWAL/AADE
Wright-Patterson AFB, OH 45433-6543

Other

Defense Technical Information Center (2 copies)
Cameron Station
Alexandria, VA 22314

CSED Review Panel

Dr. Dan Alpert, Director
Center for Advanced Study
University of Illinois
912 W. Illinois Street
Urbana, Illinois 61801

Dr. Barry W. Boehm
TRW Defense Systems Group
MS 2-2304
One Space Park
Redondo Beach, CA 90278

Dr. Ruth Davis
The Pymatuning Group, Inc.
2000 N. 15th Street, Suite 707
Arlington, VA 22201

Dr. Larry E. Druffel
Rational Machines
1501 Salado Drive
Mountain View, CA 94043

Mr. Neil Eastman, Manager
IBM, Financial Applications Systems
Bldg. 929, Room 1C12
21 First Field Road
Gaithersburg, MD 20878

Dr. C.E. Hutchinson, Dean
Thayer School of Engineering
Dartmouth College
Hanover, NH 03755

Mr. Oliver Selfridge
45 Percy Road
Lexington, MA 02173

IDA

General W.Y. Smith, HQ
Mr. Seymour Deitchman, HQ
Mr. Robin Pirie, HQ
Ms. Karen H. Weber, HQ
Dr. Jack Kramer, CSED
Dr. Robert I. Winner, CSED (5 copies)
Dr. John Salasin, CSED
Dr. Joe Linn, CSED (5 copies)
Ms. Katydean Price, CSED (2 copies)
IDA Control & Distribution Vault (3 copies)

END

2-87

DTIC